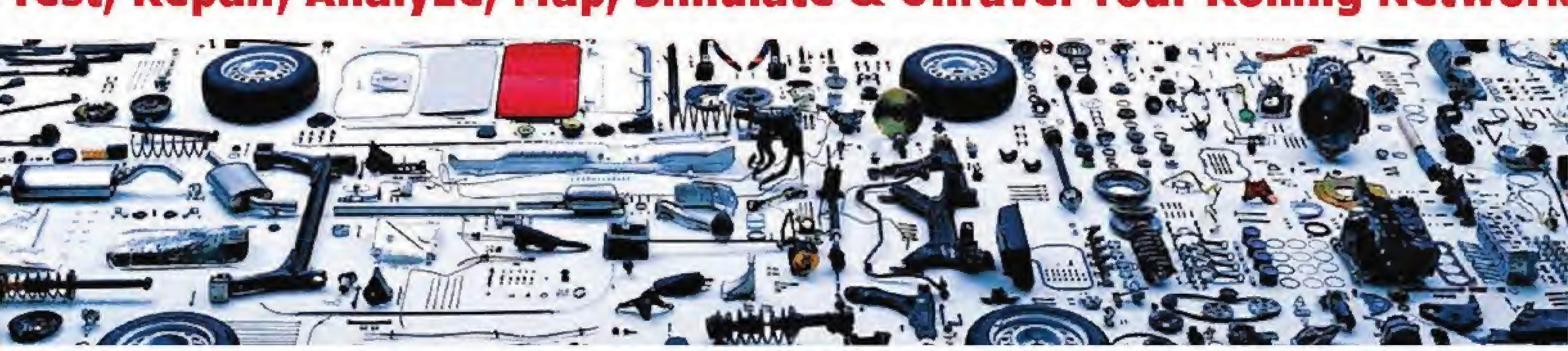
elektor .magazine
November 2013

CAN Tester

Test, Repair, Analyze, Map, Simulate & Unravel Your Rolling Network



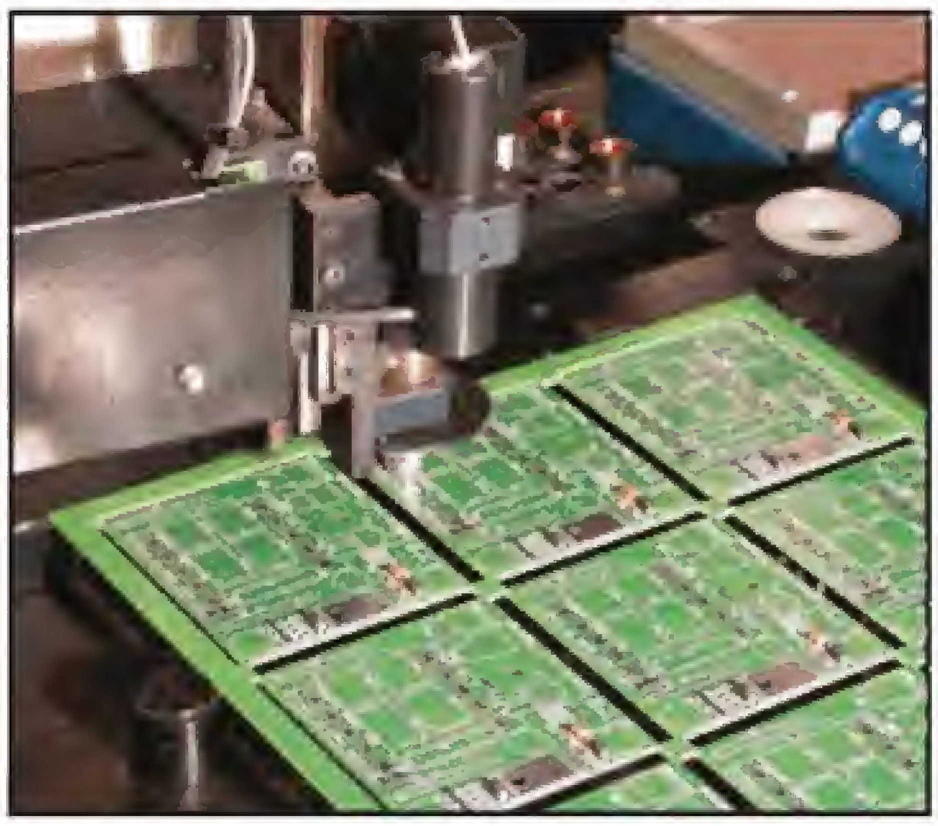
● **Linux Board Mk. 2** | LCR Meter V 3.0 | **Multichannel Temperature Logger**
USB Thermometer | Bidirectional 4-Channel Audio Selector | LED Lighting for Model Buildings ● **Standards for Coding** | FPGA Pins Afloat | PCB Milling Master Class ● 1935 Audio Spectrometer

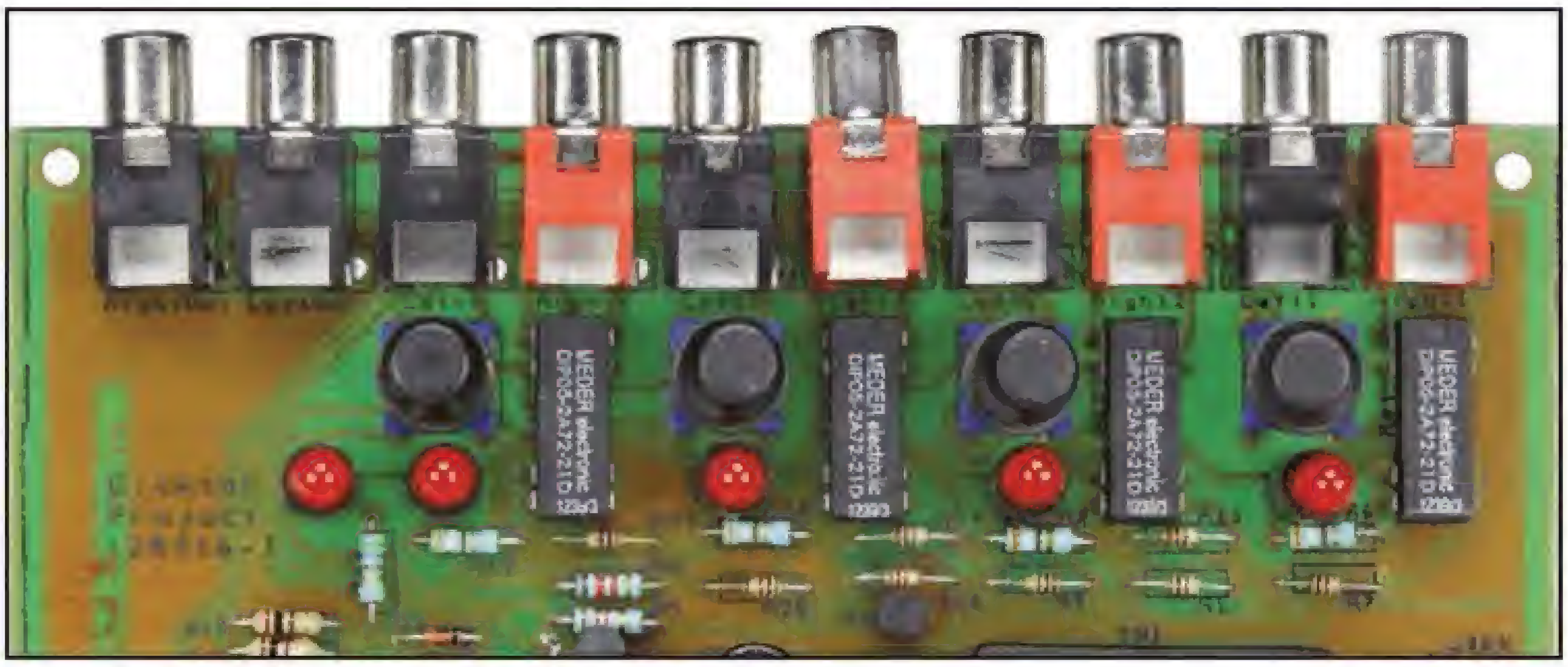
US $9.00 - Canada $10.00
11
7 25274 24965 7

FTDI Chip
AVAILABLE NOW
DISPLAY MADE EASY
EVE
EMBEDDED VIDEO ENGINE
A REVOLUTIONARY SOLUTION ENABLING HI-QUALITY, HUMAN MACHINE INTERFACES AT A LOWER COST
DEVELOPMENT SYSTEMS FROM
$35
DISPLAY AUDIO TOUCH
FTDI FT800
WATCH EVE DEMOS ONLINE
FTDICHIP.COM

Contents

elektor •magazine

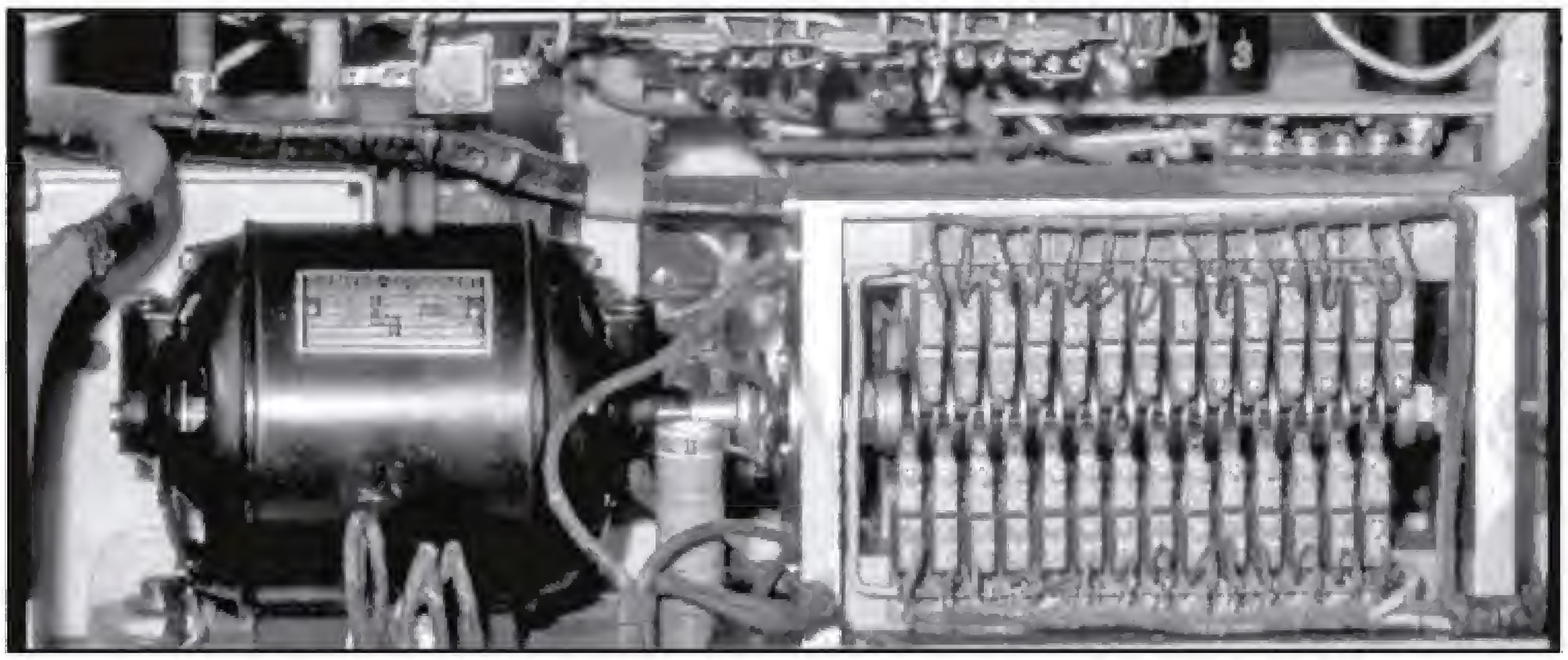


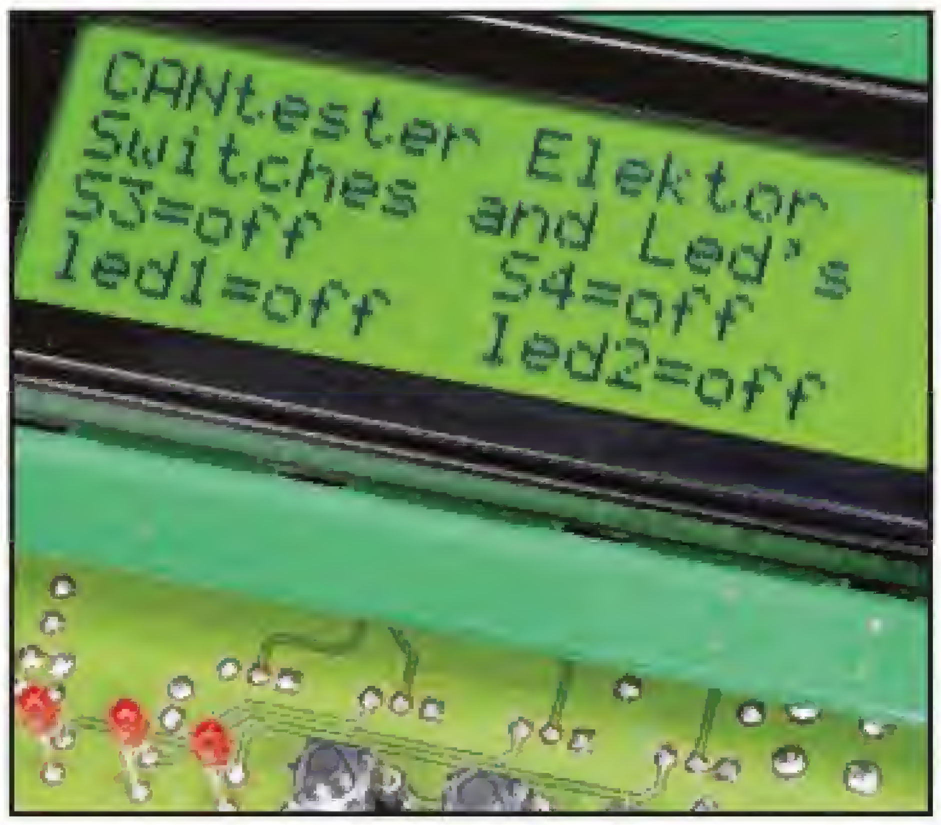


Community

Projects

Volume 39 - No. 443 November 2013





Volume 39, No. 443
November 2013

ISSN 1947-3753 (USA / Canada distribution)
ISSN 1757-0875 (UK / ROW distribution)
www.elektor.com

Elektor Magazine is published 10 times a year including double issues in January/February and July/August, concurrently by

Elektor International Media
111 Founders Plaza, Suite 300
East Hartford, CT 06108, USA
Phone: 1.860.289.0800
Fax: 1.860.461.0450

and

Elektor International Media
78 York Street
London W1H 1DP, UK
Phone: (+44) (0)20 7692 8344

Head Office:
Elektor International Media b.v.
PO Box 11
NL-6114-ZG Susteren
The Netherlands
Phone: (+31) 46 4389444
Fax: (+31) 46 4370161

USA / Canada Memberships:
Elektor USA
P.O. Box 462228
Escondido, CA 92046
Phone: 800-269-6301
E-mail: elektor@pcspublink.com
Internet: www.elektor.com/members

UK / ROW Memberships:
Please use London address
E-mail: service@elektor.com
Internet: www.elektor.com/member

USA / Canada Advertising:
Peter Wostrel
Phone: 1.978.281.7708
E-mail: peter@smmarketing.us

UK / ROW Advertising:
Johan Dijk
Phone: +31 6 15894245
E-mail: j.dijk@elektor.com

www.elektor.com/advertising
Advertising rates and terms available on request.

Can the CAN

Not so long ago you'd be greeted by rust, holes, grime, goo, mushrooms or mice droppings when removing a panel or a cover from a broken down car in order to get access to "the electrics".
Today you are bound to encounter electronics of the black box or disposable type. Removing rust or corrosion is a tough but rewarding job that actually helps to get the car on the road again. By contrast, remove any piece of reasonably advanced electronics from a post 2000 vehicle and you may be unable to even switch on the ignition. In-vehicle electronic systems like OBD, ECU and CAN require a total change of mind in terms of repair and maintenance work. There's no denying that these systems are hugely successful and the way forward, but bear in mind that cars with *zero-electronics-on-board-except-the-radio* are increasingly popular too.
The best way to explore the route and workings of a bus is to buy a ticket for a round trip and be kind to the driver. In this edition we present a tool that should help to remove any fears of addressing issues in vehicles that appear related to the CAN bus (if fitted!). The tool is not just analytic, but educational too as it allows a good deal of messages from CAN devices to be simulated in the comfort of your electronics lab, as opposed to a garage outfit with Orange County Choppers on the floor and Lady Gaga on the radio. The actual function of the CAN Tester board is one of about a dozen as determined by the firmware it is running.
This month we are again honoring your requests for projects related to measurement. In this case it's temperature all round with our Multichannel Temperature Logger (page 26) and the USB Thermometer (page 32). The first reads up to six sensors and writes .csv files into a spreadsheet—the second has one sensor and writes directly to the PC over a USB link.
Besides CAN and temperature measurement the pages ahead present audio, embedded Linux, PCB milling, Modeling, Coding, racing on hydrogen power, restoring a 1935 spectrometer and a hex-coded puzzle challenge. I wonder if that 260 HP Forze VI on page 68 has CAN controlled brakes? Or a radio?

Enjoy reading this edition of Elektor,

Jan Buiting, Editor-in-Chief

The Team

Editor-in-Chief:	Jan Buiting
Associate Editor:	Thijs Beckers
Publisher / President:	Hugo Van haecke
Membership Managers:	Shannon Barraclough (USA / Canada), Raoul Morreau (UK / ROW)
International Editorial Staff:	Harry Baggen, Eduardo Corral, Wisse Hettinga, Denis Meyer, Jens Nickel
Laboratory Staff:	Ton Giesberts, Luc Lemmens, Tim Uiterwijk, Clemens Valens, Jan Visser
Graphic Design & Prepress:	Giel Dols, Jeanine Opreij, Mart Schroijen
Online Manager:	Daniëlle Mertens
Managing Director:	Don Akkermans

USA
Hugo Van haecke
+1 860-289-0800
h.vanhaecke@elektor.com

United Kingdom
Don Akkermans
+31 46 4389444
w.hettinga@elektor.com

Germany
Ferdinand te Walvaart
+49 241 88 909-17
f.tewalvaart@elektor.de

France
Denis Meyer
+31 46 4389435
d.meyer@elektor.fr

Netherlands
Erik Jansen
+31 46 4389429
h.baggen@elektor.nl

Spain
Eduardo Corral
+34 91 101 93 95
e.corral@elektor.es

Italy
Maurizio del Corso
+39 2.66504755
m.delcorso@inware.it

Sweden
Don Akkermans
+31 46 4389444
w.hettinga@elektor.com

Brazil
João Martins
+31 46 4389444
j.martins@elektor.com

Portugal
João Martins
+31 46 4389444
j.martins@elektor.com

India
Sunil D. Malekar
+91 9833168815
ts@elektor.in

Russia
Nataliya Melnikova
+7 (965) 395 33 36
Elektor.Russia@gmail.com

Turkey
Zeynep Köksal
+90 532 277 48 26
zkoksal@beti.com.tr

South Africa
Johan Dijk
+31 6 1589 4245
j.dijk@elektor.com

China
Cees Baay
+86 21 6445 2811
CeesBaay@gmail.com

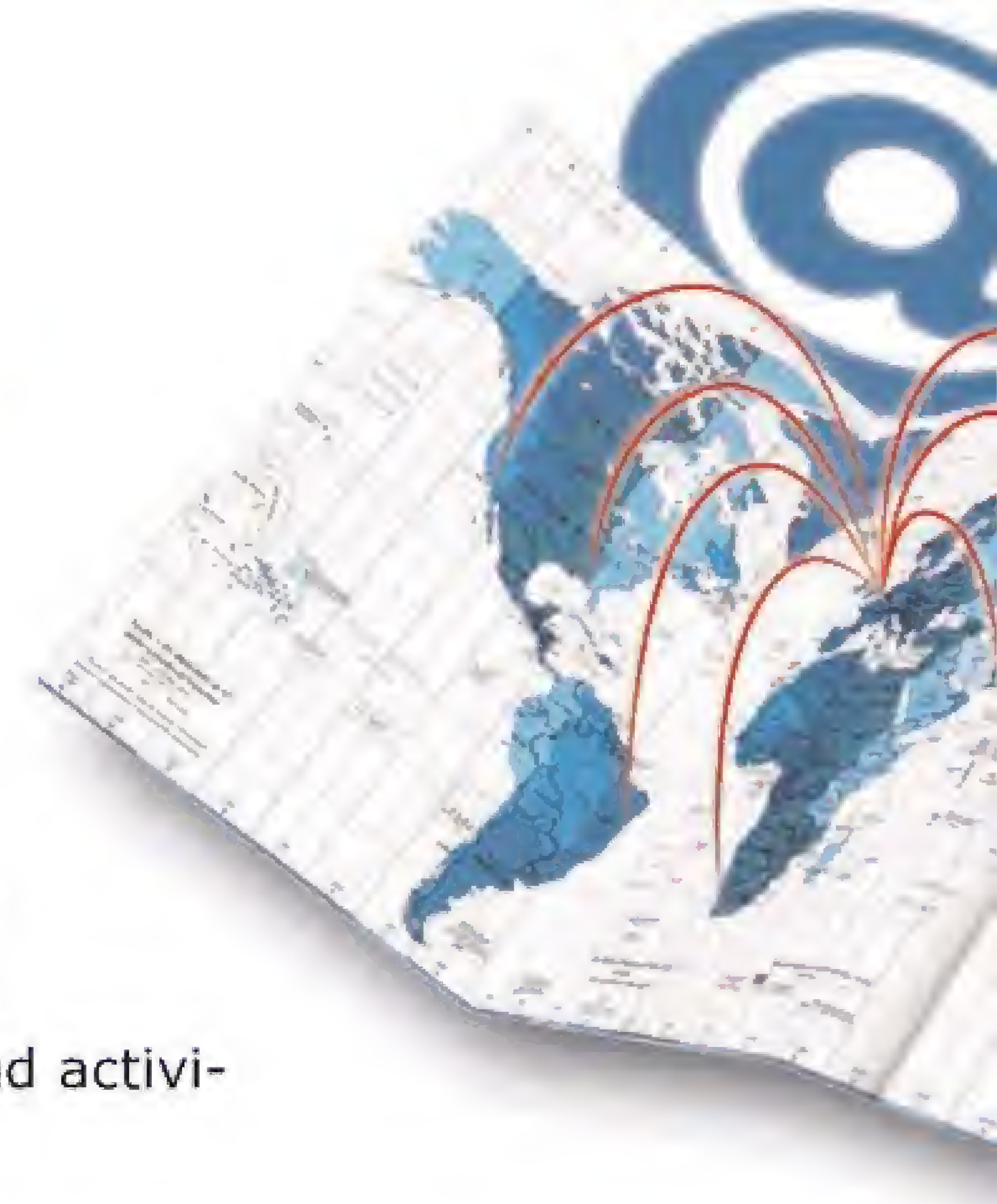
Our Network

Connects You To

Supporting Companies

Not a supporting company yet?

Contact Peter Wostrel (peter@smmarketing.us, Phone 1 978 281 7708,
to reserve your own space in Elektor Magazine, Elektor•POST or Elektor.com



Elektor World

Compiled by
Wisse Hettinga

Every day, every hour, every minute, at every given moment designers and enthusiasts are thinking up, tweaking, reverse-engineering and developing new electronics. Chiefly for fun, but occasionally fun turns into serious business. Elektor World connects some of these events and activities — for fun and business.

Let Your Brain Do The Thinking

...and the software do the work. Here at the MOSI (Museum of Science and Industry) in Manchester, UK, a group of 20 designers from a selection of companies rally to take the new DesignSpark Mechanics to the limits. The challenge is to develop something 'life saving' in 48 hours and have it 3D printed! Not an easy task for people who are normally doing many things besides saving the globe, but the ideas that come up are interesting. Currently it's fair to say that the Elektor team's world-saving activities are limited to preventing a few hapless plants from drying out in their office—it's a start.

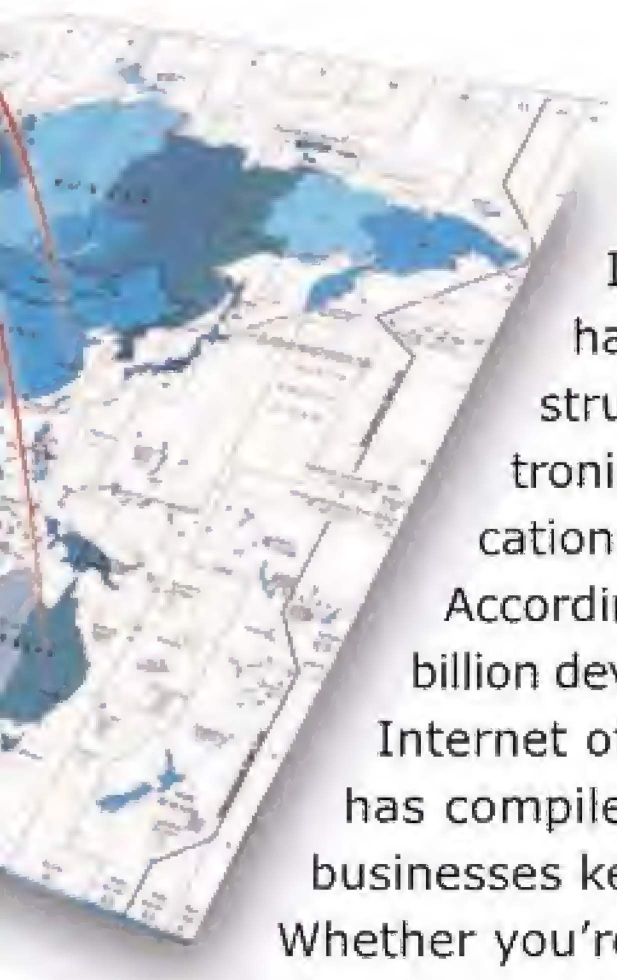
All Smiles

The guys you see smiling are on the the Flowcode designers' team. They have every reason to be happy, just having finished Flowcode version 6, a new and exciting product.

Where other coding programs let you see a blinking LED or a signal level, Flowcode 6 allows you to program and simulate a complete 3D world. Typical 3 D CAD program designs can be imported and brought to life. For example, you can have your 3D printer design first simulate and operate before you start the actual production. In this way this new version of Flowcode is bridging the gap between electronics programming and simulation of real world actions.

The team is based in Halifax, UK and I am sure we can expect more applications and developments from them in the near future.

Flowcode is available in the Elektor Store, see www.elektor.com/flowcode



The 'Internet of Things' Launching Point

It's a bold but not implausible idea that someday many everyday devices will have embedded sensors that enable them to communicate via an Internet-like structure. Consumer products from cars to household appliances and other electronics would be able to connect through such local and global networks. The implications for individuals and business models are enormous.

According to the marketing firm ABI Research, more than 30 billion devices will be connected wirelessly through the so-called Internet of Things (IoT) by 2020. And *Circuit Cellar* magazine has compiled a list of online resources to help individuals and businesses keep up with the evolution of the IoT.

Whether you're looking for a workshop in Italy on wireless sensor networks, contact information for innovators in the field, or details about new IoT applications and breakthroughs, you'll likely find helpful information online in the "Internet of Things (IoT) Resources" feature at circuitcellar.com/featured/iot-resources.

And it's a list that's expected to grow. If you know of a resource that should be added, please email it to CJ Abate and Mary Wilson on editor@circuitcellar.com.ome up with proper answers, simple and easy to understand to help you to take the next step on your favorite embedded platform. That step is called Arduino.next and will be up & running soon—powered by Elektor. Stay tuned to our communication channels!

Follow us on Facebook, www.facebook.com/arduinonext, and on Twitter, @ arduinonext, and check out the Arduino products already on sale at www.elektor.com.

That Sounds Good!

audio xpress
ADVANCING THE EVOLUTION OF AUDIO TECHNOLOGY
audioxpress.com

In the audio electronics domain, Elektor International Media publishes *audioXpress*, *Voice Coil*, *Loudspeaker Industry Sourcebook*, *World Tube Directory*, books, and more. Those titles were founded in the US by Edward T. Dell (1923-2013) and for over 35 years served the do-it-yourself audio constructor as well as those working in the audio industry with great articles, projects, tips and technologies.

Believing that the work of enthusiasts should serve as a model for the industry as far as excellence of design and quality of constructions goes, Ed Dell launched *The Audio Amateur* in 1970, a magazine devoted exclusively to DIY audio. Ten years later, believing there was sufficient interest in the loudspeaker market, Ed launched a separate magazine in 1980, called *Speaker Builder*, while a third publication, *Glass Audio*, responded to the increasing interest in vacuum tube based audio equipment. In 1996, *Audio Amateur* was renamed *Audio Electronics* and in 2000, the three magazines were combined into a single, monthly periodical, named *audioXpress*. In 2011, Ed Dell sold his company to Elektor International Media.

A new editorial team, reinforced by leading authors from the Elektor network, is currently working on a redesign of the publication with an expanded format, addressed towards the global audio engineering community, covering also the R&D efforts in the industry in many new application areas.

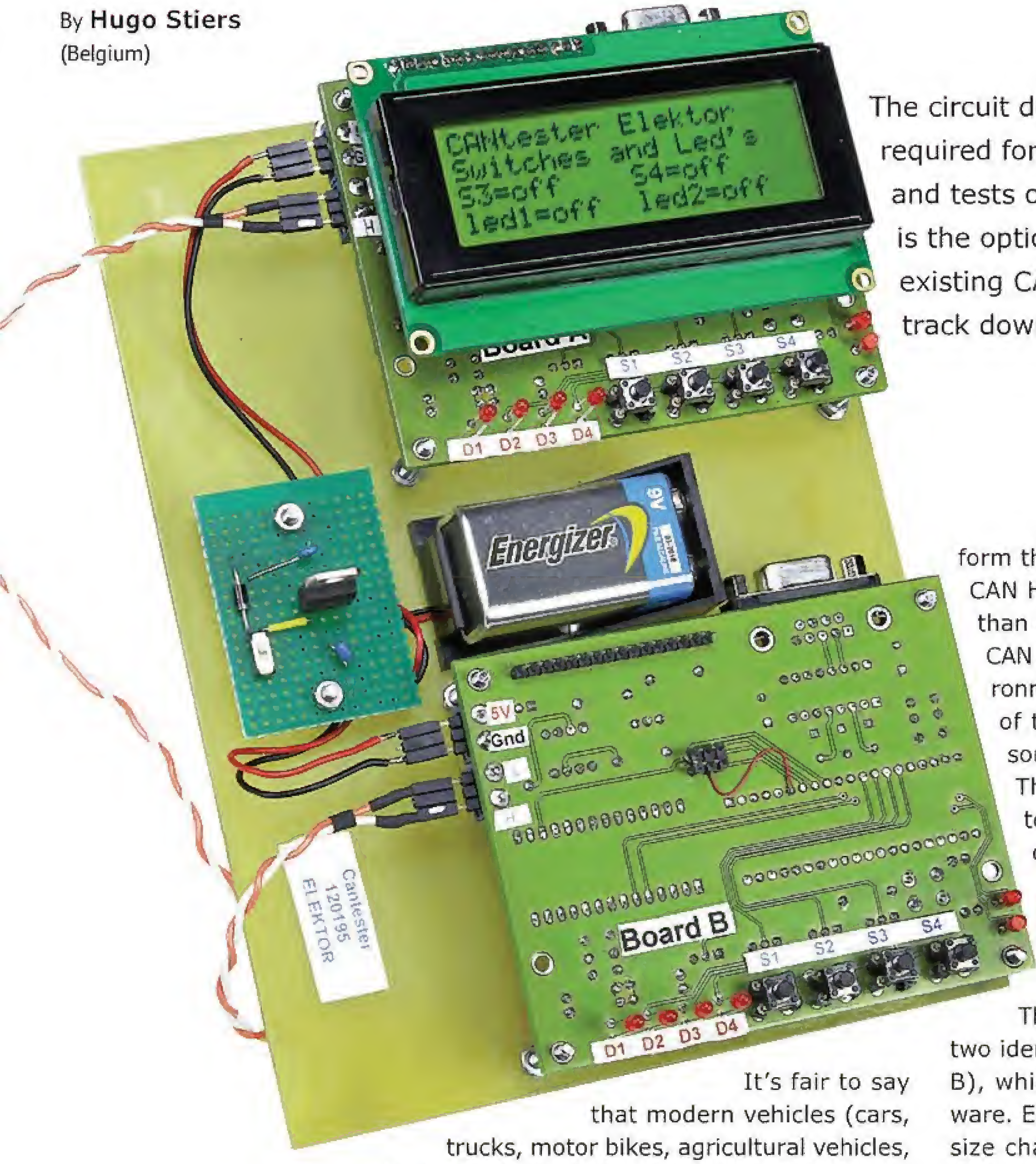
The refreshed, restyled *audioXpress* will be launched at the AES Convention in NY (October 17th – 20th) with a new graphic layout in print and in full digital front, including a regular newsletter to over 30,000 members (at the time of writing).

audioXpress is already engaging with the global audio community though Twitter (@audioXP_editor) and Facebook (facebook.com/audioxpresscommunity). See more at www.audioxpress.com.

CAN Tester

With comprehensive features

By **Hugo Stiers**
(Belgium)



The circuit described here has all the features required for conducting various experiments and tests on a CAN bus. In addition there is the option of connecting the tester to an existing CAN bus to monitor the data or to track down faults.

It's fair to say that modern vehicles (cars, trucks, motor bikes, agricultural vehicles, etc.) have these days become rolling (mobile) networks. The various control systems in these vehicles are connected together with a network used for exchanging messages. In this way it is ensured that the various functions in these vehicles are functioning optimally.

Many car manufacturers use the CAN bus (Controller Area Network) for this. The control units are connected together with two twisted wires (terminated with resistors at the ends) and so form the CAN network. These wires are called CAN High and CAN Low. There may be more than one CAN network in a single vehicle.

CAN is a system that works reliably in an environment with high interference. But because of the complexity of CAN networks it can sometimes be difficult to solve problems. This is one of the reasons why the Elektor CAN Tester has been developed; the other reason is that the CAN Tester is also excellent to gain experience with the CAN bus and offers the possibility of experimenting with software for CAN circuits.

The CAN Tester described here comprises two identical circuit boards (board A and board B), which are only loaded with different software. Each board can be equipped with a 4x20 size character LCD. The boards communicate with each other according to the CAN protocol. These boards can also be connected to an existing CAN bus.

The CAN Tester offers the following options (using the same boards):

- Test configuration with boards A and B, for 29-bit and/or 11-bit IDs (automatic);
- Reading out of CAN data on the LCD (for example parking brake, odometer reading, etc.);

- Examine data using HyperTerminal;
- Test functionality with pushbuttons and LEDs;
- Simulation of messages.

These features will all be discussed in this article.

The hardware

We start with a brief description of the hardware that has been used. In **Figure 1** this is shown for one circuit board, the other board has an identical design.

The circuit consists of the following components:

- ATmega8515: 8-bit microcontroller (IC1);
- SJA1000: CAN protocol controller (IC4);
- PCA82C250: CAN transceiver (IC3);
- MAX232: RS232 transceiver (IC2, for communication with the PC);
- 4 x 20 character LCD (LCD1).

The SJA1000 is a bitstream processor with a transmit and receive buffer. This is controlled and initialized by the ATmega8515. The ATmega8515 provides the transmit buffer of the SJA1000 with messages and reads out the receive buffer. The SJA1000 is connected to the ATmega8515 via a multiplexed address/data bus (PA0 – PA7).

In addition, there are four control signals that go to the SJA1000: CS (chip select), ALE (address latch enable), RD (read) and WR (write). CS (chip select) has to be logic Low when the ATmega8515 communicates with the SJA1000. The ALE signal has to be logic High when an address is on the bus, and logic Low for data. The RD and WR signals are used to determine whether it is a Read or Write command from or to the memory in the SJA1000.

The interrupt output (INT) of the SJA100 is not used here. With the mode connection input pin of

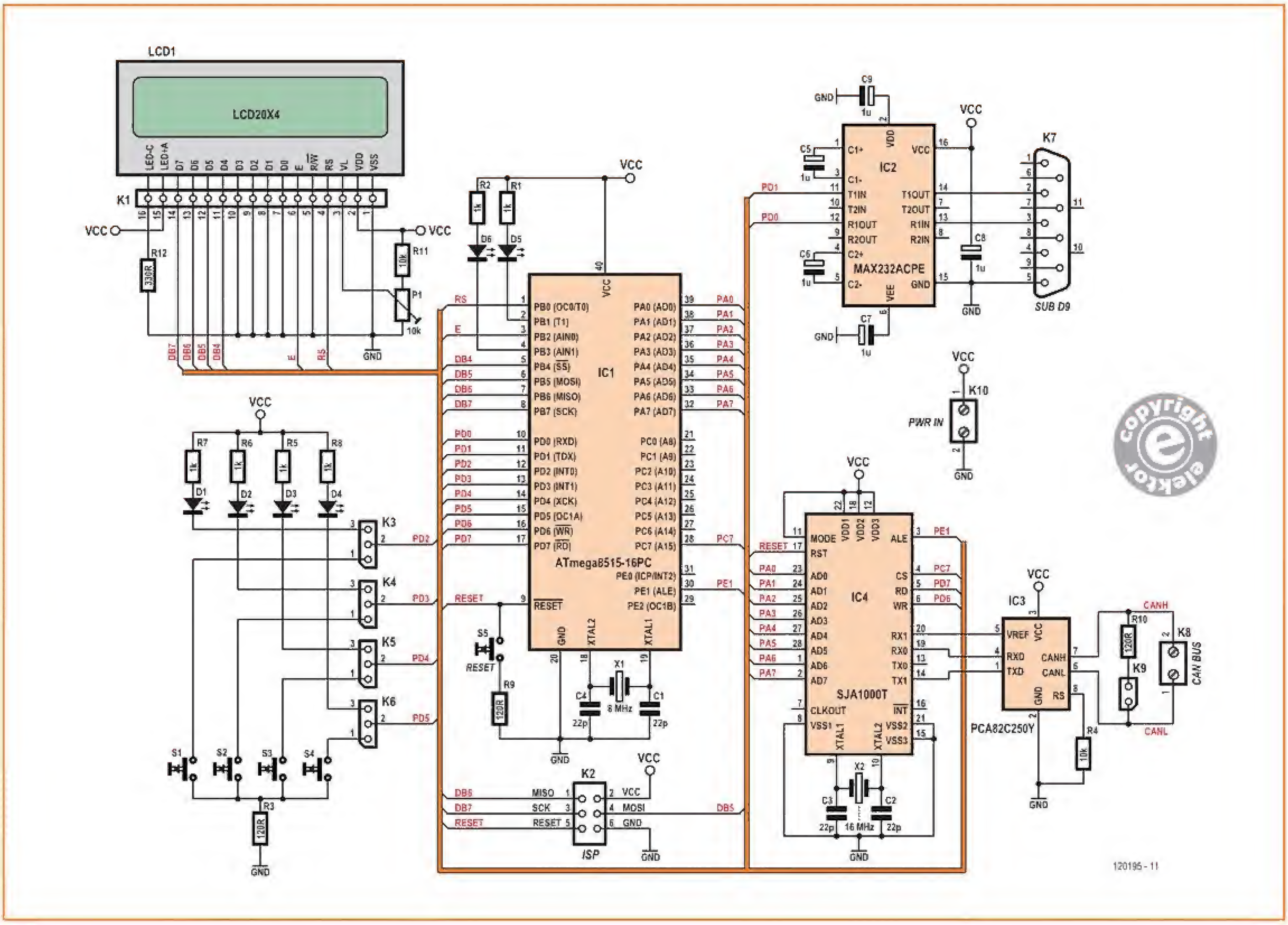


Figure 1.
Schematic for the CAN Tester. The main ingredients are a microcontroller, a CAN protocol controller and a CAN transceiver.

the SJA1000, the bus interface can be configured for a microcontroller made by Intel or one made by Motorola. With this CAN tester that is Intel—that is why pin 11 of IC4 is connected to V_{CC}.
The ATmega8515 sees the SJA1000 as an expansion of its (internal) RAM. This is the reason why in the compiler settings for Bascom the 'EXTERNAL ACCESS ENABLE' has to be ticked (see the box 'Program settings').
The PCA82C250T transceiver ensures that the data it receives on its TXD pin (TTL level) is converted to the differential signal (as a difference voltage) of the CAN bus (CanH and CanL, with the Can-High and Can-Low wires as a twisted pair, terminated with two resistors of 120 Ω). The received differential data is converted by the transceiver to a signal with TTL levels, which goes from its RXD connection to the SJA1000.
The ATmega8515 runs at a clock frequency of 8 MHz, the SJA1000 at 16 MHz. The baud rate with the serial connection to the PC operates at a speed of 57,600 baud.
The board is fitted with four jumpers (K3 through K6), which are used to select whether the LEDs or the pushbuttons are connected to port D of the ATmega8515.
The display is used in 4-bit mode and is connected to port B of the microcontroller. P1 is used to adjust the contrast for the display.
The MAX232 is an old acquaintance; it provides for the conversion of the 5-V signals on the board to the 12-V signals of the RS232 bus.
K9 is used to enable the termination resistor for the CAN bus.
In addition there is a 6-way ISP connector to allow the microcontroller to be programmed while in the board. You can, for example, connect the STK500 programmer to this.
The entire circuit is powered from 5 V. You can use a wall adapter with a regulated output for this or a 9-V battery with a separate voltage regulator. The current consumption is small, for short-duration experiments a battery will be sufficient.

The software

The software is written in BASCOM (demo version). The ATmega8515 was programmed with the STK500 (Atmel). This software is based on the examples from BASCOM (third party Lawicel). This software contains the minimum of what is required to send and receive data frames (messages). The software configures the SJA1000 in the PELICAN mode. In this mode you can send and receive 11-bit and 29-bit identifiers.
The software comprises seven parts:

1. First an address is issued to the registers of the SJA1000 (since the ATmega8515 sees the SJA1000 as an external RAM expansion).
2. An identifier (29 bits) is turned into a 'Long' (4 bytes), with an 11-bit identifier this becomes a 'Word' (2 bytes).
3. The 'Do Loop' contains the actions that the program will carry out. From here the subroutines *Transcantest1*, *Transcntest2* and *Receivecantest1* are called.
4. The subroutine 'Initsja' is used to initialize the SJA1000, this contains, among other things, the setting for the bit rate.
5. The subroutines *Transcantest1* and *Transcantest2* ensure that the data frames (messages) are transmitted.
6. The subroutine *Receivecantest1* is responsible for the reception of the data frames. This subroutine also contains what has to be done with the received data (processing by the ATmega8515).
7. Display of the data on the LCD (4-bit mode).

The bit-rate of the SJA1000 is here set to 250 Kbits/s (the same as the J1939 standard). Other bit-rates can be set in the software, taking into account the clock frequency of the SJA1000 (16 MHz). You can find various 'bit rate calculators' on the Internet for the SJA1000, which will give you the values for the registers (tmg_0 and tmg_1).
Further explanation of how the software is put together can be found in the data sheets and the application notes for the SJA1000 (with respect to the registers in this IC).
The software contains comments that provide further explanation for certain program lines.
In a separate Word document that you can download from [1], you will find an overview of the features of the available software together with some explanation.

Construction

In **Figure 2** we can see the circuit board that has been designed for the CAN Tester. This is fitted with parts on both sides. Most of the components are fitted on the side with the component overlay. On the solder side are the LEDs D1 through D6, the pushbuttons S1 through S4 and the 16-way header for the LCD. The circuit was originally designed for ICs with 'normal' pins, but in the meantime two of the ICs used here are now only available in SMD version: the PCA82C250 (IC3) and the SJA1000 (IC4). In order to be able to use these on the existing circuit board we have used small adapter boards (available from [2], among others). For those who buy the preprogrammed controller we will also supply the two adapter boards with it, so that you can get started immediately.

The LCD is not necessary for all of the test configurations. It all depends on the firmware used (see also the additional documentation available as a free download [1].

There is a sub-D9 connector for the connection to the PC. You can, if necessary, connect a USB/RS232 adapter cable to this for communicating with a modern computer.

All the firmware is of course available as a free

COMPONENT LIST

Resistors

R1,R2,R5–R8 = 1kΩ
R4,R11 = 10kΩ
R3,R9,R10 = 120Ω
R12 = 330Ω
P1 = 10kΩ trimpot, e.g. Bourns 3386P-1-103LF, Newark / Farnell # 9355030

Capacitors

C1–C4 = 22pF
C5–C9 = 1µF 63V radial

Semiconductors

D1–D6 = LED, red, 3mm
IC1 = ATmega8515-16PC, programmed, Elektor Store # 120195-42a for board A, # 120195-42b for board B
IC2 = MAX232ACPE
IC3 = PCA82C250 (8-pin DIP) or PCA82C250T (SO8, adapter board required)
IC4 = SJA1000 (28-pin DIP) or SJA1000T (SO28, adapter board required)

Miscellaneous

X1 = 8MHz quartz crystal
X2 = 16MHz quartz crystal
LCD1 = LCD, 4x20 characters (Elektor Store # 120061-73)
K1 = 16-pin pinheader, 0.1'' pitch
K2 = 6-pin (2x3) pinheader, 0.1'' pitch
K3–K6 = 3-pin pinheader, 0.1'' pitch, with jumper
K7 = 9-way sub-D socket, right angled pins, PCB mount
K8,K10 = 2-way PCB screw terminal block, 0.2'' pitch
K9 = 2-pin pinheader, 0.1'' pitch, with jumper
S1–S5 = miniature pushbutton with make contact, e.g. TE Connectivity 3-1437565-0, Newark / Farnell # 2060813
PCB # 120195-I, see [1]

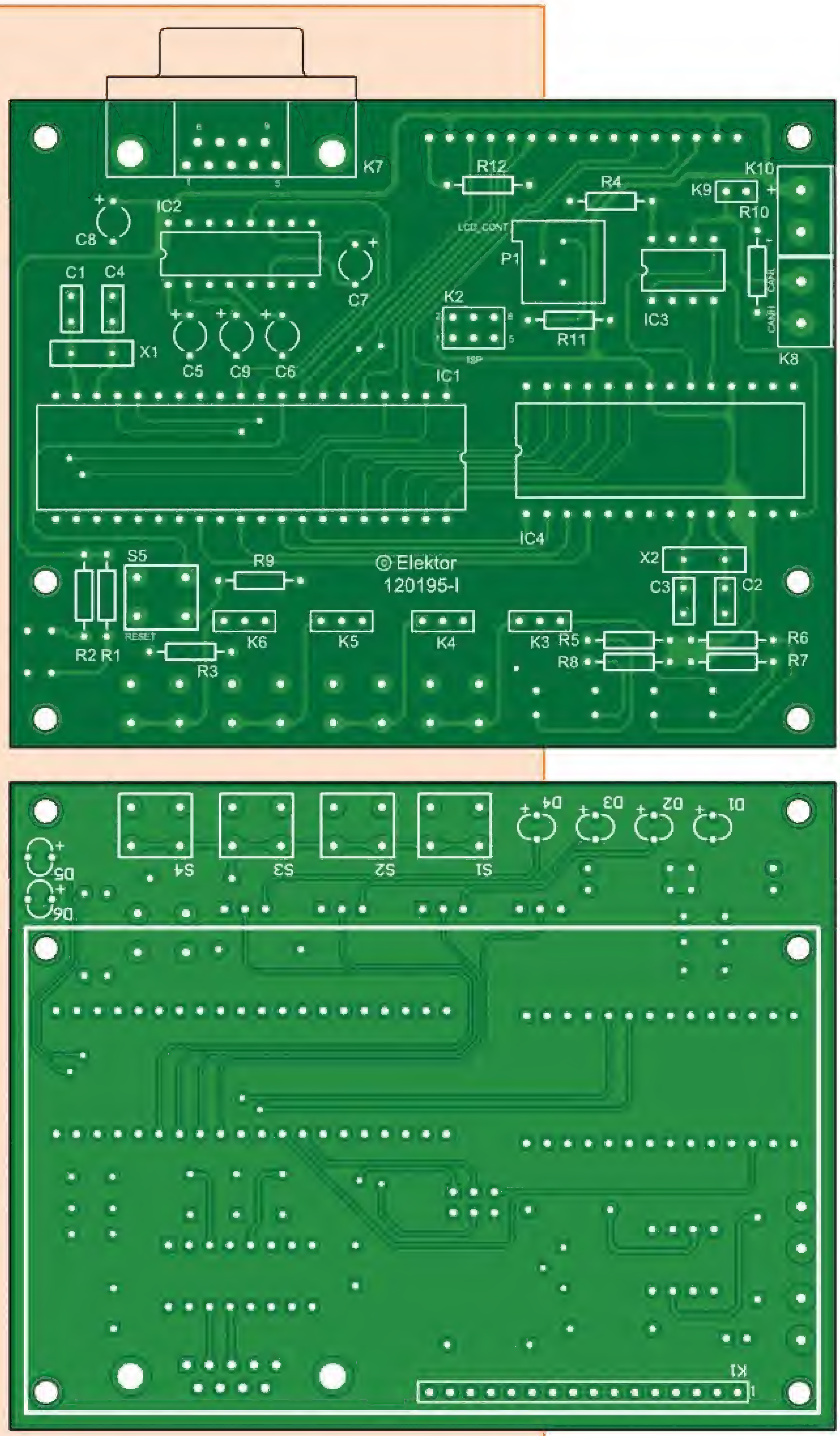
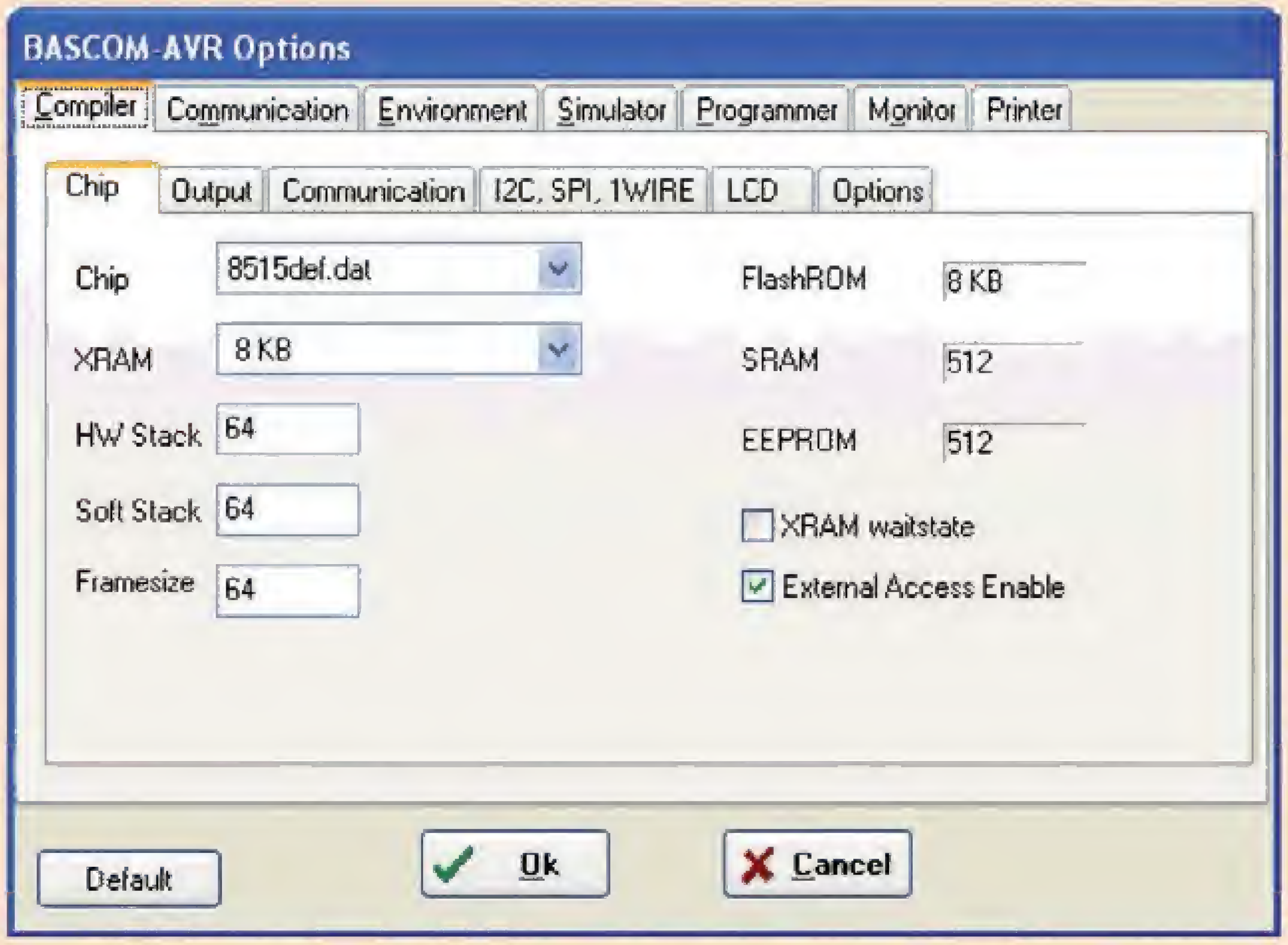


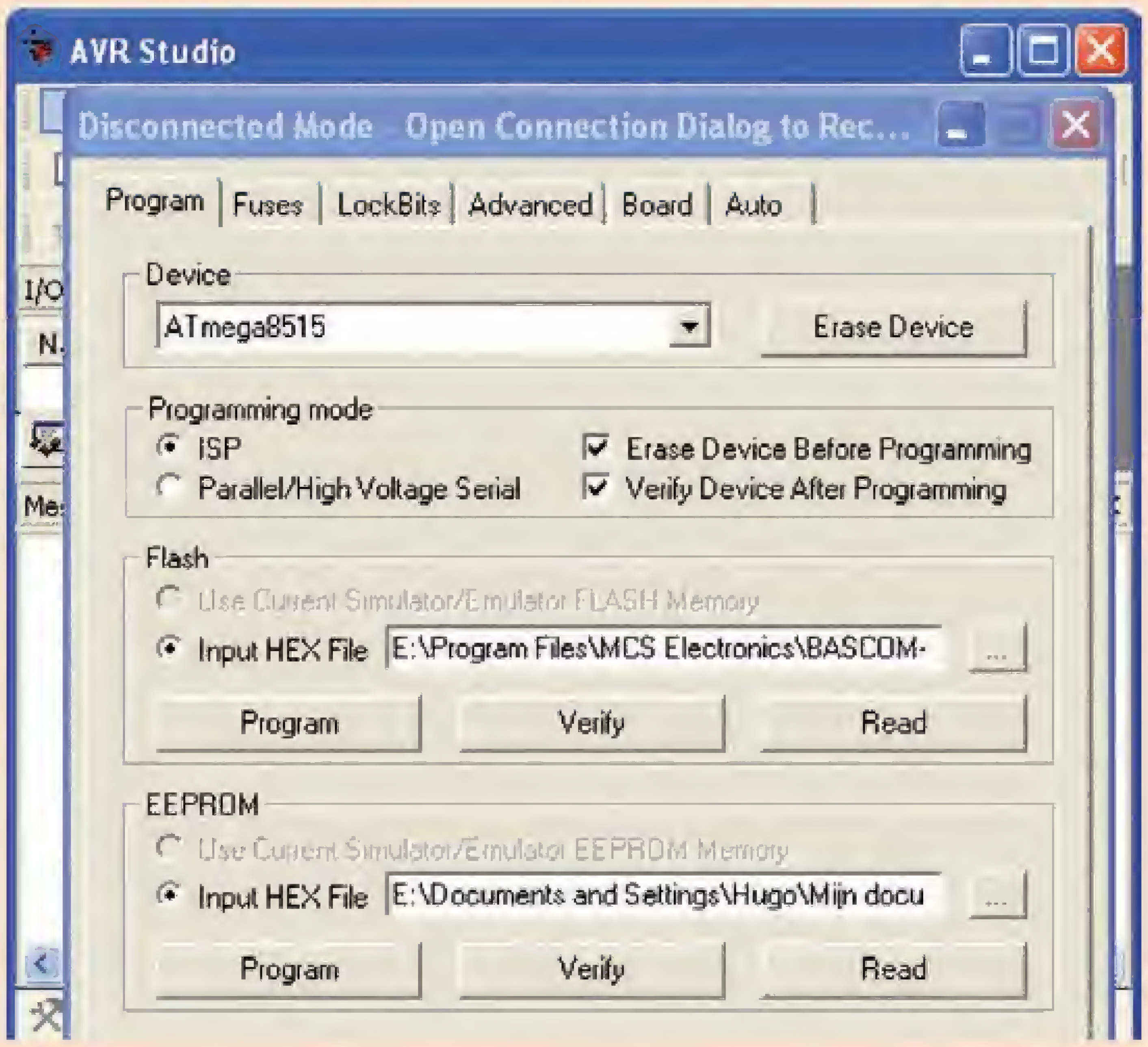
Figure 2.
The circuit board contains components on both sides: on one side the LEDs, pushbuttons and the display, on the other side all the other parts.

Program settings

In BASCOM the compiler has to be configured by selecting under: OPTIONS/Compiler/C 'External Access Enable'.



The settings for AVR Studio 4 together with the STK500 are as follows:



The fuses are set in AVR Studio as follows:

Boot Flash section size = 128 Boot start address =$0F80; BOOTZ = 11
Brown-out detection level at VCC = 2.7V; (BODLEVEL = 1)
Ext. Crystal/Resonator High Freq.; Start-up time: 16K CK+64ms;(CKSL = 1111 SUT = 11)

download from the Elektor website [1].
Each application requires a different firmware. In the interest of simplicity, Elektor Store only supplies the preprogrammed microcontrollers for application 4, which is described a little further on (120195-42a and b).

Applications for the CAN Tester

Here follows a brief description of the various applications, where each time also the necessary firmware versions are mentioned.

Application 1: Board A and board B send and receive messages automatically to each other

firmware: 120195-40a (board A without LCD)
120195-40b (board B without LCD)
120195-41a (board A with LCD)
120195-41b (board B with LCD)

Here, both boards send and receive messages to each other with 29-bit IDs. Each message contains 8 data bytes, of which only one data byte used.
Board A transmits messages that are only intended for board B, and board B sends messages that are only destined for board A. Messages are received the same way: Board A only received messages from board B and the other way around.
The transmitted data byte appears on port D of the microcontroller and is made visible on both boards using the four LEDs, which turn on and off two at a time. This also signals that there is continuous data traffic between the two boards. That therefore also means that the wiring between the boards is correct.

You can use this function only to test the wiring of a CAN network. You can connect the boards to any arbitrary point on the network wiring. You connect the boards (A and B) to the ends of that section of wiring that you would like to test. Take into account any termination resistors that are on the CAN network already (and if necessary disconnect them), each board of the CAN-tester has a termination resistor of 120 Ω, which you can switch in or out with jumper K9 (the bus impedance is 60 Ω).

With this setup you can test:

- interruption of CanH;
- interruption of CanL;

Consider your safety!

Know what you are doing! When the CAN Tester is connected to a vehicle and you send messages (data frames) on the network, then it is possible for engines to start automatically, vehicles starting to move by themselves, engines to reach high RPM, etc. Take the time to work safely, don't endanger yourself and others. Closely follow the instructions from the manufacturer and the vehicle. And read the SAFETY instructions.

- CanH and CanL swapped;
- CanH and CanL shorted;
- Moisture in the cables (plugs submerged in water).

When any of these faults appear or are present, then the LEDs will stop flashing immediately. When the fault disappears the LEDs will start to flash again. In this way you will have a visual indication of a fault. To track down intermittent faults you can shake the wires and plugs about while at the same time keeping an eye on the CAN Tester.

The CAN Tester works optimally on an inactive network, in this case the CAN bus is entirely available to the CAN Tester. It also works on an active network, but the LEDs will flash slower in this case because there will also be other data traffic on the bus. When the LEDs flash that means the messages from the boards are sent and received in between the other messages.

Application 2: Single CAN Tester with LCD (handbrake, odometer reading, etc.)

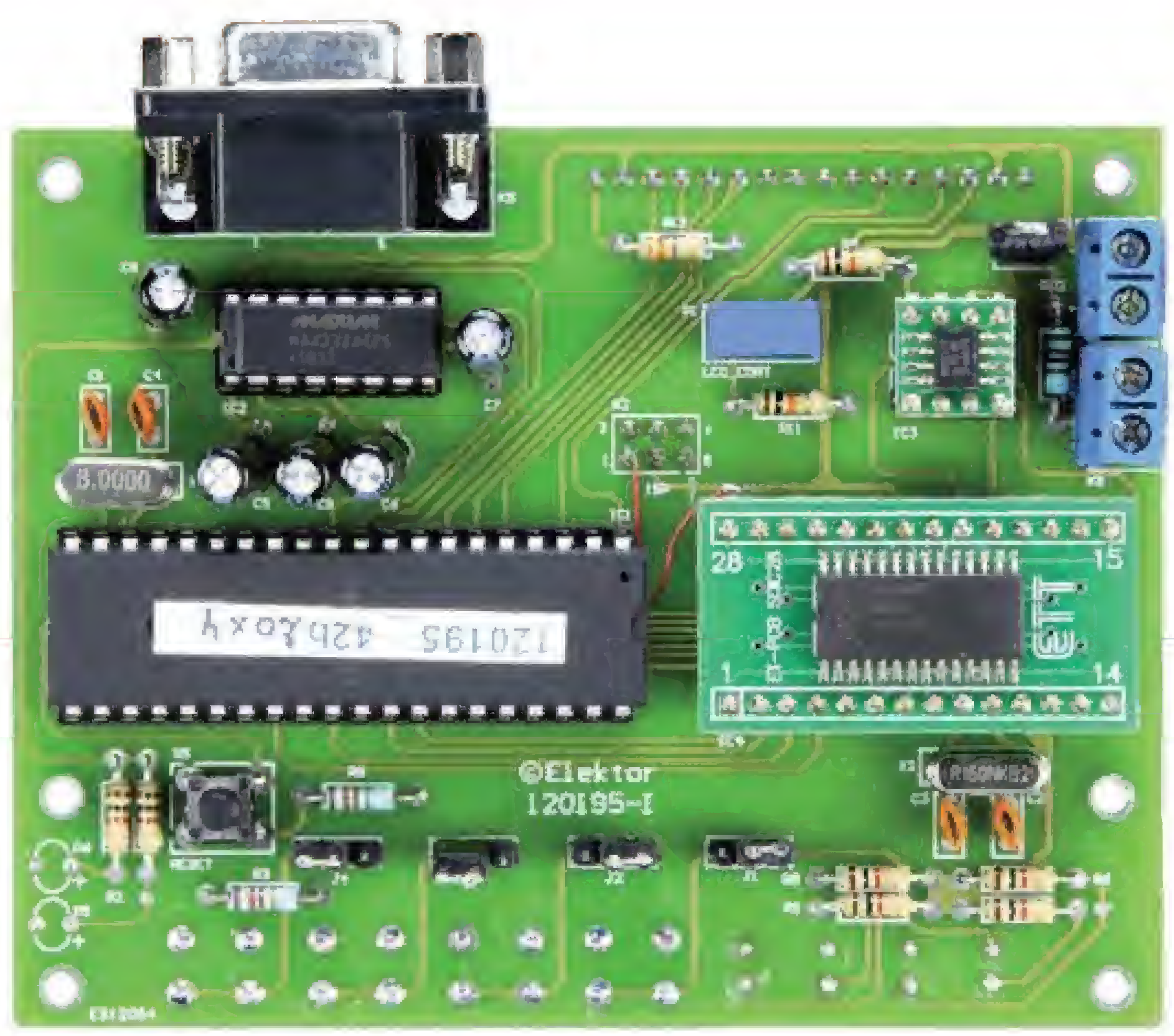
Firmware: 120195-44a (board with LCD, handbrake)
120195-44b (board with LCD, odometer reading)
120195-45a (board with LCD, acceleartor)

In this application the CAN Tester only receives messages. These are displayed intelligibly on the 4x20 character LCD.

The three examples are:
a) status of the handbrake of a truck;
b) the odometer (miles counter) reading;
c) accelerometer position.

These examples show how you can process the received data into a legible result with the aid of a few operations. This can also be used for diagnostics, for example if you would like to read a certain sensor during a test drive.

In this application the CAN Tester is connected



to the network of a vehicle that transmits these messages.
If you do not have a vehicle available to you, then you can also simulate these messages with another board. To simulate messages you can, for example, use the Tiny-CAN View (see Automotive CANtroller, Elektor February 2009 or the CAN Explorer, Elektor February 2008).

Application 3: Viewing data using HyperTerminal (baudrate = 57,600)

This is possible with all version of the firmware The CAN Tester has a MAX232 for communicating with a PC. The software is written in such a way that we can examine the contents of messages on a PC via the serial port. This applies to both the sender and the receiver.
The received data can also be stored in a file (via the HyperTerminal program). This can be all messages, or only those that are of interest to you. You can set that yourself in the software. For example, each program contains the part number of the software that is in the microcontroller at the time. This is very handy when you are using multiple controllers (running different firmware). By connecting them to a PC you can see which program it contains.

Application 4: CAN Tester with pushbuttons and LEDs

Firmware: 120195-42a (board A with LCD)
120195-42b (board B with or without LCD)
This application requires two boards (Board A and Board B).
The messages have 29-bit identifiers and the bitrate is 250 Kbits/s (J1939 protocol).
We use two pushbuttons and two LEDs. Place the jumpers in the correct positions for this:
- Ports D4 and D5 to the pushbuttons (jumpers K5: 1-2 and K6: 1-2).
- Ports D2 and D3 to the LEDs (jumpers K3: 2-3 and K4: 2-3).

Both boards (A and B) can receive and transmit messages.

Board A:
A message is sent on both the press and the release on one or both pushbuttons. This transmitted message causes the LEDs on board B to turn on or off (LEDs turn on when pressing the pushbuttons and turn off when released). Only one data byte is sent, with identifier 0C1F134A(H).
Board A receives only messages from board B, which consist of one data byte with identifier 0C1F1315(H).

Board B:
A message is sent on both the press and the release on one or both pushbuttons. This causes one or both LEDs on board A to turn on or off (LEDs turn on when pressing the pushbuttons and turn off when released). Only one data byte is sent, with identifier 0C1F1315(H).
Board B receives only messages from board A, which consist of one data byte with identifier 0C1F134A(H). The received data byte contains the information of what the LEDs should do.

When receiving the messages the identifier has to correspond with the one that is mentioned in the software, otherwise the data is not copied to port D. Board A accepts only the data from board B and the other way around. The LCD shows the state of the pushbuttons and LEDs.
You can use the CAN Tester in this application in various ways, for example when testing the wiring of a CAN network. When operating the pushbuttons on board A the LEDs on board B have to follow these pushbuttons, and the other way around.

This configuration and software has also been tested in an active network (which also carries other messages). This works, but is slower (always check which identifiers you use, these may not be the same as those already in the active network).
In this application too, the CAN Tester works best when the wiring of the network does not contain any other activity.

Application 5: The CAN Tester as a simulator for messages

Firmware: 120195-43a (board A with LCD, see additional documentation 120195-W).
120195-43b1 (board B with LCD, see additional documentation 120195-W).

With this firmware it is possible to simulate messages; you program messages in one board and use the other board to show them on the LCD or display them via HyperTerminal. These messages can be transmitted automatically with a certain repetition frequency (repetition time) on the CAN bus. You can also transmit them when

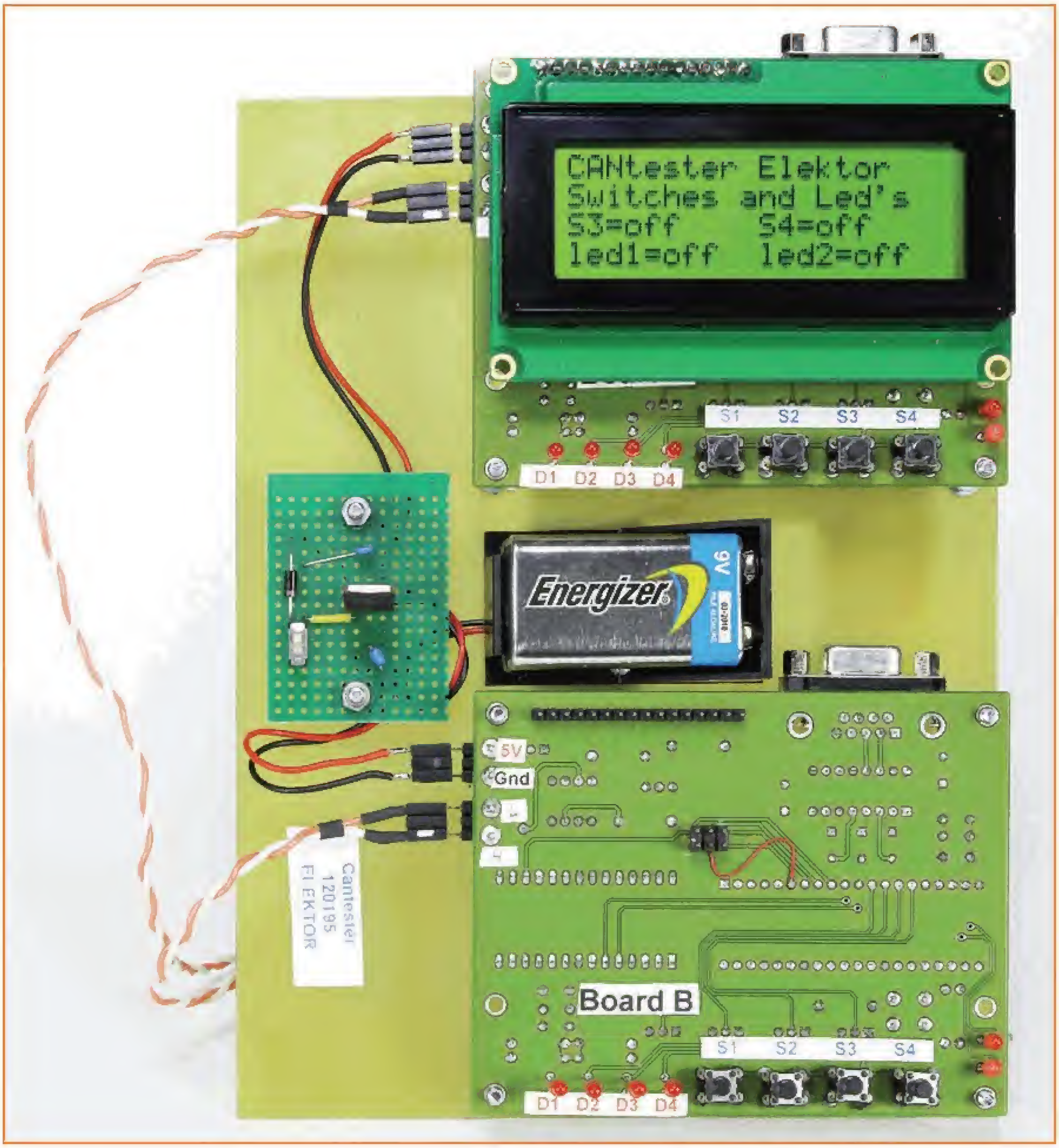


Figure 3.
The test setup for application #4. The supply voltage is provided by a 9-V battery and a 7805 voltage regulator.

operating the pushbuttons.
Firmware 120195-43a: sending messages; firmware 120195-43b1: receiving messages.
Examples and explanation of messages are at [3].
To test the boards (for displaying data on the LCD), you can again use Tiny CAN View or use the CAN Explorer.

Finally

The CAN Tester always has to be connected to another board, vehicle or other test setup such as Tiny CAN View or the CAN Explorer from Elektor. We wish you much success with your tests and experiments. (120195-I)

Internet Links

[1] www.elektor.com/120195

[2] www.futurlec.com/SMD_Adapters.shtml

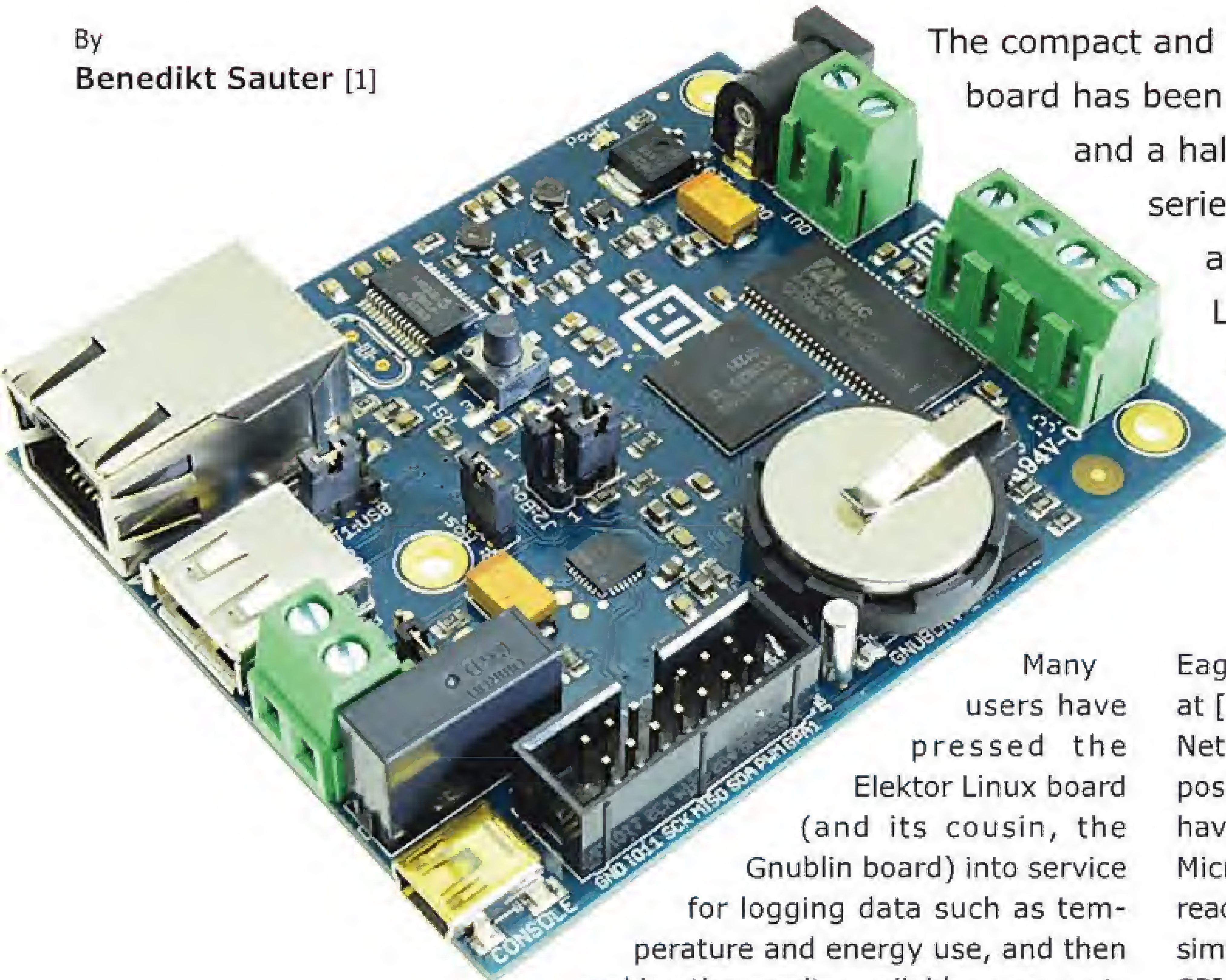
[3] www.fms-standard.com/down_load/fms_document_ver02.00vers_11_11_2010.pdf

Elektor Linux Board: New and Improved!

Now with LAN and real-time clock

By
Benedikt Sauter [1]

The compact and low-cost Elektor Embedded Linux board has been available to buy for around a year and a half now. Along with its accompanying series of articles it offers even beginners access to the world of embedded Linux. It is now time for us to update the board based on feedback from the user community.



Many users have pressed the Elektor Linux board (and its cousin, the Gnublin board) into service for logging data such as temperature and energy use, and then making the results available over a network. Hence it is clear that a built-in network interface and real-time clock would be useful additions for an update to the board (Figure 1). We have also incorporated other suggestions made by users: for example, the fixing holes are bigger, so that the board can be mounted more easily.

An extra serving of chips

The basic circuit of the board has not changed since the first version [2]. Alongside the processor we see 32 MB of RAM, a CP2102-based USB-to-serial adapter, and the power supply circuitry. Also familiar from the first Elektor Linux board will be the 14-way expansion connector, which allows a wide range [3][4] of extension boards (again available from Elektor [5]) to be connected. The circuit diagram and printed circuit board layout can be downloaded from the Elektor website in Eagle format; a free Eagle viewer is available at [7], and a good book on Eagle at [12].

Network access is provided using a special-purpose device, as the LPC3131 processor does not have its own integrated Ethernet hardware. The Microchip ENC28J60 [8] will be known to some readers as a network adapter popular for use with simple 8-bit processors. It is connected using an SPI port and an interrupt signal. A suitable driver for this device has been available in the kernel archive for some time.

The real-time clock (RTC) device chosen is the MCP7940 [9], which requires an external crystal. If a coin cell is added to provide back-up power the RTC will continue to keep time even when the board is switched off.

Configuring the network

The first step in using the ENC28J60 LAN device on the board is the command

```
modprobe enc28j60 irq_pin=12 cs_pin=19
```

should show the device as interface 'eth0'.

In order to have the board receive an IP address from a DHCP server elsewhere on the network, use the command:

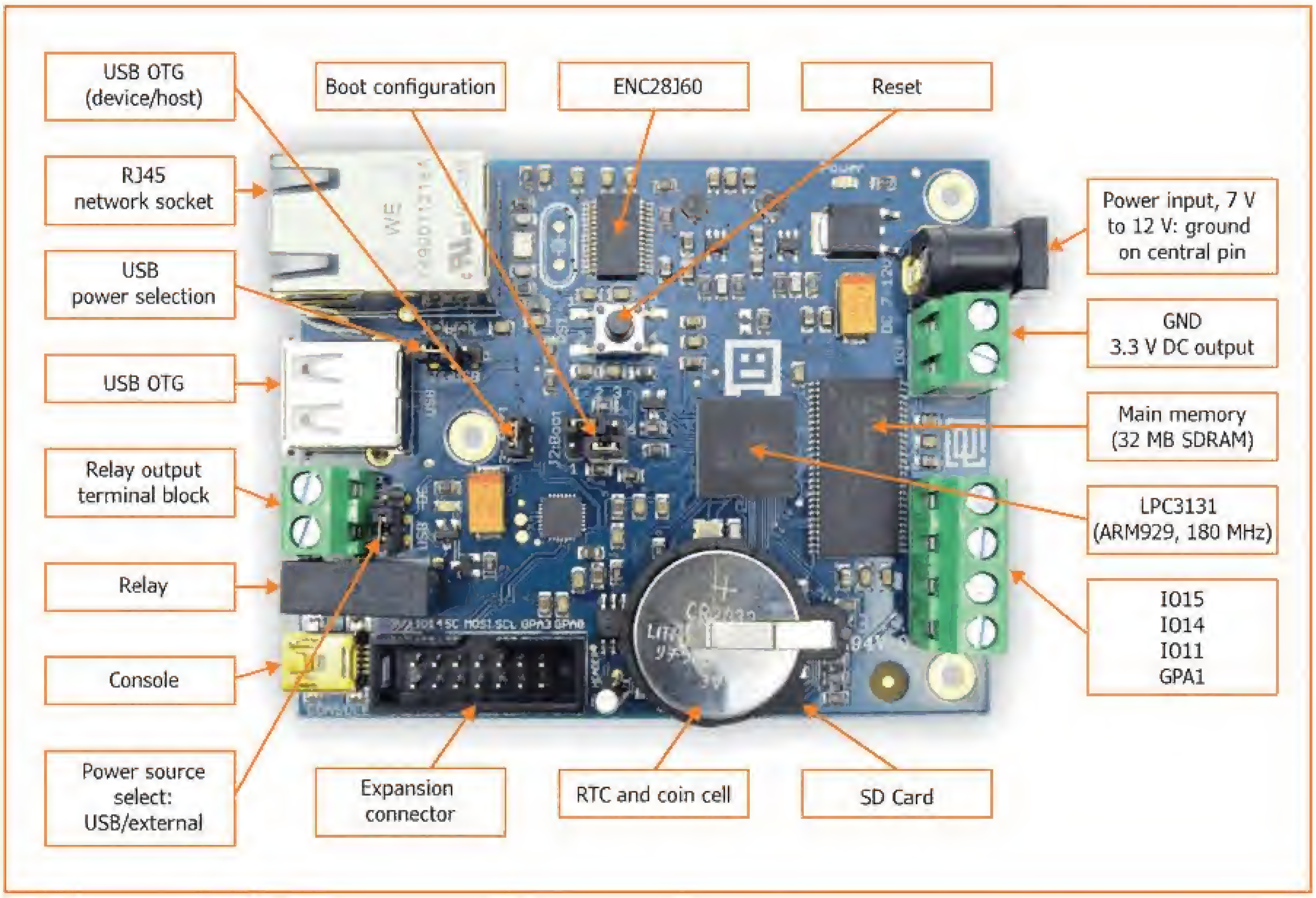


Figure 1. The new version of the board includes a network interface and a real-time clock.

```
ifconfig -a
```

should show the device as interface 'eth0'.
In order to have the board receive an IP address from a DHCP server elsewhere on the network, use the command:

```
dhclient eth0
```

We can now test the interface by pinging any other machine on the network or server on the Internet:

```
ping google.de
```

The output should appear as shown in Figure 2. Stop the 'ping' program in the usual way by pressing control-C.

Loading the driver automatically

If you would like the ENC28J60 driver to be loaded automatically when the system boots up, add the line

```
enc28j60 irq_pin=12 cs_pin=19
```

to the file '/etc/modules' in the board's file system. This is most easily done from the console. The command

```
echo "enc28j60 irq_pin=12 cs_pin=19" >> /
etc/modules
```

will append the given line to the end of the file. Alternatively, the file can be edited using the 'nano' text editor:

```
nano /etc/modules
```

Using a fixed MAC address

If the board is being issued with a different IP address by the DHCP server each time it is booted the reason is likely to be that the ENC28J60 does not contain a fixed MAC address: each time the driver is loaded it is configured with a different address. A fixed MAC address can be specified to get around this problem. Add the following line in the file '/etc/network/interfaces':

```
hwaddress ether MAC-ADDRESS
```

A suitable choice might be the one given to the LAN module the first time its driver is loaded. This can be determined using the command

```
ifconfig
```

where it is displayed as the 'Hwaddr' as follows:

```
eth0      Link encap:Ethernet  HWaddr
```

```
ba:07:1b:0c:64:00
```

Real-time clock

A simple command is all that is needed to set the clock:

```
gnublin-rtc -s "2013/01/20 11:23:12"
```

To read back the time from the device, use the command:

```
gnublin-rtc -g
```

To set the Linux system clock from the RTC, use the command:

```
gnublin-rtc -x
```

Having the Linux clock automatically set from the RTC on each boot-up takes a little more effort. First add the following text in the file '/etc/rc.local', just before the line that reads 'exit 0':

```
echo mcp7940 0x6f > /sys/bus/i2c/devices/
i2c-1/new_device
echo "Now setting the date and time."
sleep 1
hwclock --hctosys
```

Second, add an entry to the file '/etc/modules':

```
rtc-mcp7940
```

And finally, deactivate the hwclock shell script, as this can cause problems with this type of real-time clock device:

```
update-rc.d hwclock remove && update-rc.d
hwclock.sh remove
```

On the next reboot the board will set the system time from the hardware clock. More information on this subject can be found on the Gnublin wiki [10].

The future

A new version of the Gnublin installer is available [11], which allows a Linux PC to be used to create an SD card including bootloader, kernel and file system. The new version offers the choice between an 8 MB image and a 32 MB image.
This is accompanied by a change to the underlying file system: EXT4 is now used, which is practically fail-safe in the event of loss of power. This avoids the need for time-consuming file system checks required to 'repair' the SD card.
We will look further at this and more in an article in the next issue.

(130214)

Internet Links

[1] sauter@embedded-projects.net

[2] www.elektor.com/120181

[3] www.elektor.com/120596

[4] www.elektor.com/130212

[5] www.elektor.com/gnublin

[6] www.elektor.com/130214

[7] www.cadsoftusa.com/download-eagle/freeware

[8] http://ww1.microchip.com/downloads/en/DeviceDoc/39662A.pdf

[9] http://ww1.microchip.com/downloads/en/DeviceDoc/22266D.pdf

[10] http://en.gnublin.org/index.php/RTC_DS1307

[11] http://en.gnublin.org/index.php/Gnublin_Installer

[12] www.elektor.com/eaglestarterguide


```
PING google.de (173.194.69.94) 56(84) bytes of data.
64 bytes from bk-in-f94.1e100.net (173.194.69.94): icmp_req=1 ttl=49 time=44.3 ms
64 bytes from bk-in-f94.1e100.net (173.194.69.94): icmp_req=2 ttl=49 time=44.7 ms
64 bytes from bk-in-f94.1e100.net (173.194.69.94): icmp_req=3 ttl=49 time=43.3 ms
64 bytes from bk-in-f94.1e100.net (173.194.69.94): icmp_req=4 ttl=49 time=43.4 ms
--- google.de ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 43.308/43.968/44.784/0.673 ms
```

Figure 2. Results displayed by a successful 'ping' command: we are on the Internet!

Important update to the Elektor 500 ppm LCR Meter

By
Jean-Jacques Aubry
(France)

Since this project was published in three installments [1], several hundred of the instrument have been built, to the great satisfaction of their users, the author, and Elektor editors alike. The author has now corrected two shortcomings liable to interfere with proper operation under certain conditions.

The two points worthy of attention are the fact that the instrument seems to 'hang up' when measuring low resistances (<1 Ω); and the apparent impossibility to perform Trim actions. The author has already commented and acted on these problems in the Elektor forums [2], [3], but it's useful to revert to them here.

When measuring resistances with a value lower than 1 Ω, in order to obtain an adequate measuring voltage, the firmware sets the "**voltage**" **measurement gain** to maximum:

- range 1 (R_{sense} = 100 Ω and PGA103 gain at 100);
- final amplification gain close to maximum (step E or F).

Unfortunately, the input offset voltage is strongly amplified as well, and using the circuit published in the March 2013 edition, this is no longer compensated. This can lead to the maximum voltage at the input to the analog/digital converter being exceeded all the time. This erratic phenomenon is not systematic and depends to a great extent on the cumulative offset voltage of U6 and U4, and on the (low) resistance (or inductance) being measured. So that's the first one!

As for the second, when there is no component connected during the 'TRIM - OPEN-CIRCUIT' operation, here too the '**current**' **measurement gain** is maximum at the frequency of 100 Hz (or 120Hz):

- range 8 (R_{sense} = 100 kΩ and PGA103 gain at 100);
- final amplification gain close to maximum (step E or F).

At this point, the 'current' measuring circuit is very sensitive to interference picked up by the measuring leads, particularly from the power line (50 Hz or 60 Hz). The result of the measurement is so erratic that the firmware refuses to display it, and does not validate the "TRIM". So much for the second one!

Two problems—two solutions

Depending on whether you are handy with a soldering iron or not, there are two solutions:

- Modify the hardware—the best solution—to enable the input **offset voltage** compensation circuit to be adapted, and at the same time update the firmware (v. 3.0.0) along with the AU2011 program (v. 3.0.0). The software will automatically detect the hardware modification at runtime, due to the presence of a resistor on the P2.2 port line (LCD_SI) (**Figure 1**).
- Update with the same versions of the firmware (v. 3.0.0) and AU2011 program (v. 3.0.0), without modifying the hardware.

Important: Updating the software to version 3.0.0 must be done in either case. The new version is equally compatible with the original hardware and with the modified version described below.

Modifying the hardware

As originally designed, input offset voltage compensation is achieved by injecting a current into U6's input. Unfortunately, the result is too dependent on the (DC) resistance of the DUT.

The new circuit (**Figure 1**) applies the correction voltage at the output of U5 (INA128), and hence the DUT impedance no longer has any effect; this also makes it possible to separate the compensation for the 'current' and 'voltage' measurements. To achieve this, U5's pin 5 is no longer connected to analog ground, but to a software-adjustable voltage via a low-value resistor.

There are four steps to the modification:

- remove R34 to disable the original compensation. R42 and C35 are no longer used and can also be removed;
- replace R46 by a 10 Ω resistor (prefera-

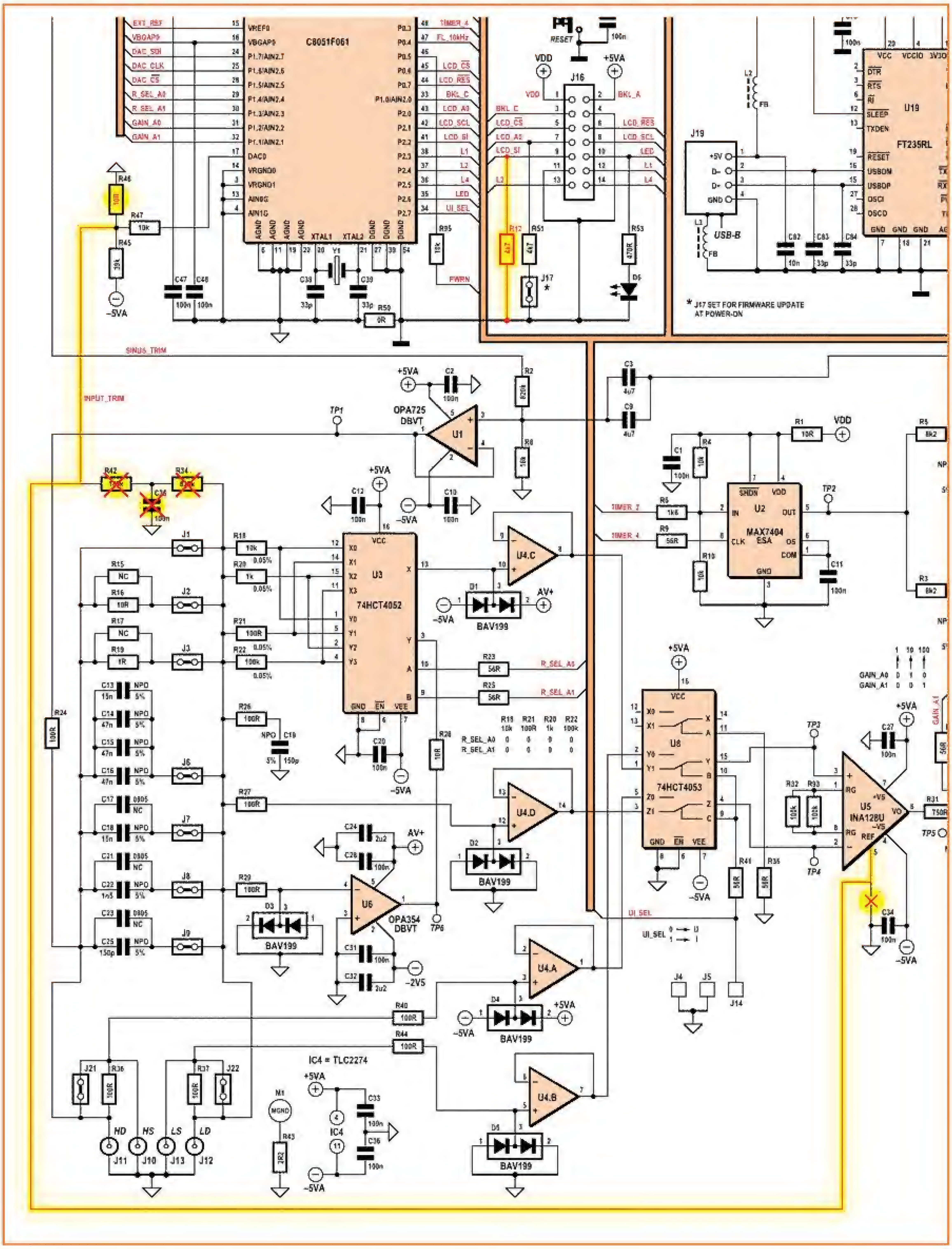


Figure 1.
Partial schematic with corrections to be made to improve the 500 ppm LCR Meter.

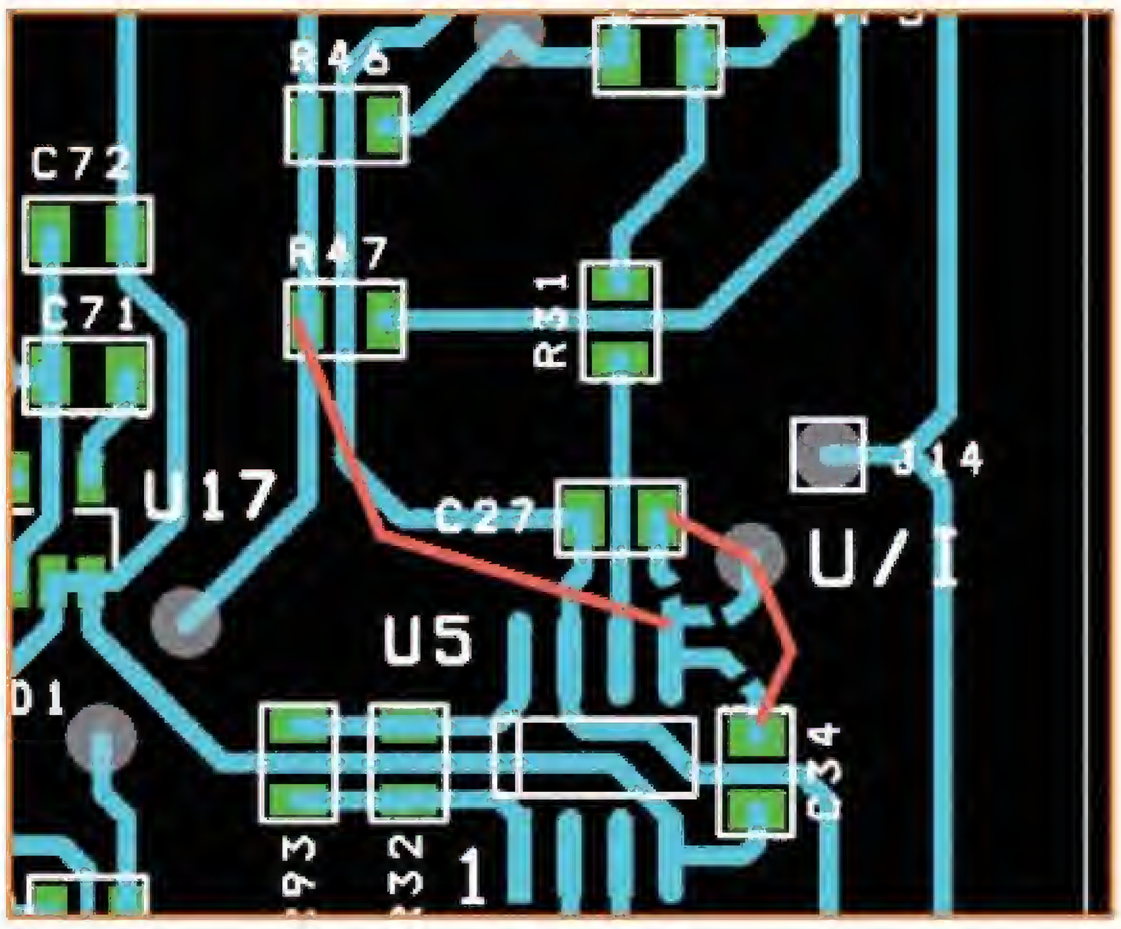


Figure 2.
Modifications to the offset voltage compensation circuit around U5.

bly 0805). The 10-Ω resistor can also be soldered directly across the 680-Ω resistor that's already there;

- the third step is more tricky: U5's pin 5 must be isolated from its present connections and connected to the junction of R45, R46, and R47. It's possible to do this by lifting pin 5, but **we advise against this, as you should not take the risk of damaging U5**. So we suggest instead making a clean cut in the (visible) tracks between U5.5 and C27, U5.5 and C34, and U5.5 and the through-hole adjacent to J14 (**Figure 2**). You'll then need to reconnect C27 and C34 to the through-hole (analog ground) via small wires (**Figure 3**); and use a short wire to create the new link between U5 pin 5 and the junction of R45, R46, and R47;
- solder a 4.7 kΩ to 10 kΩ resistor between pins 9 and 11 of J17, on the opposite side of the PCB (**Figure 4**). The presence of this resistor will allow the firmware to detect the modified circuit.

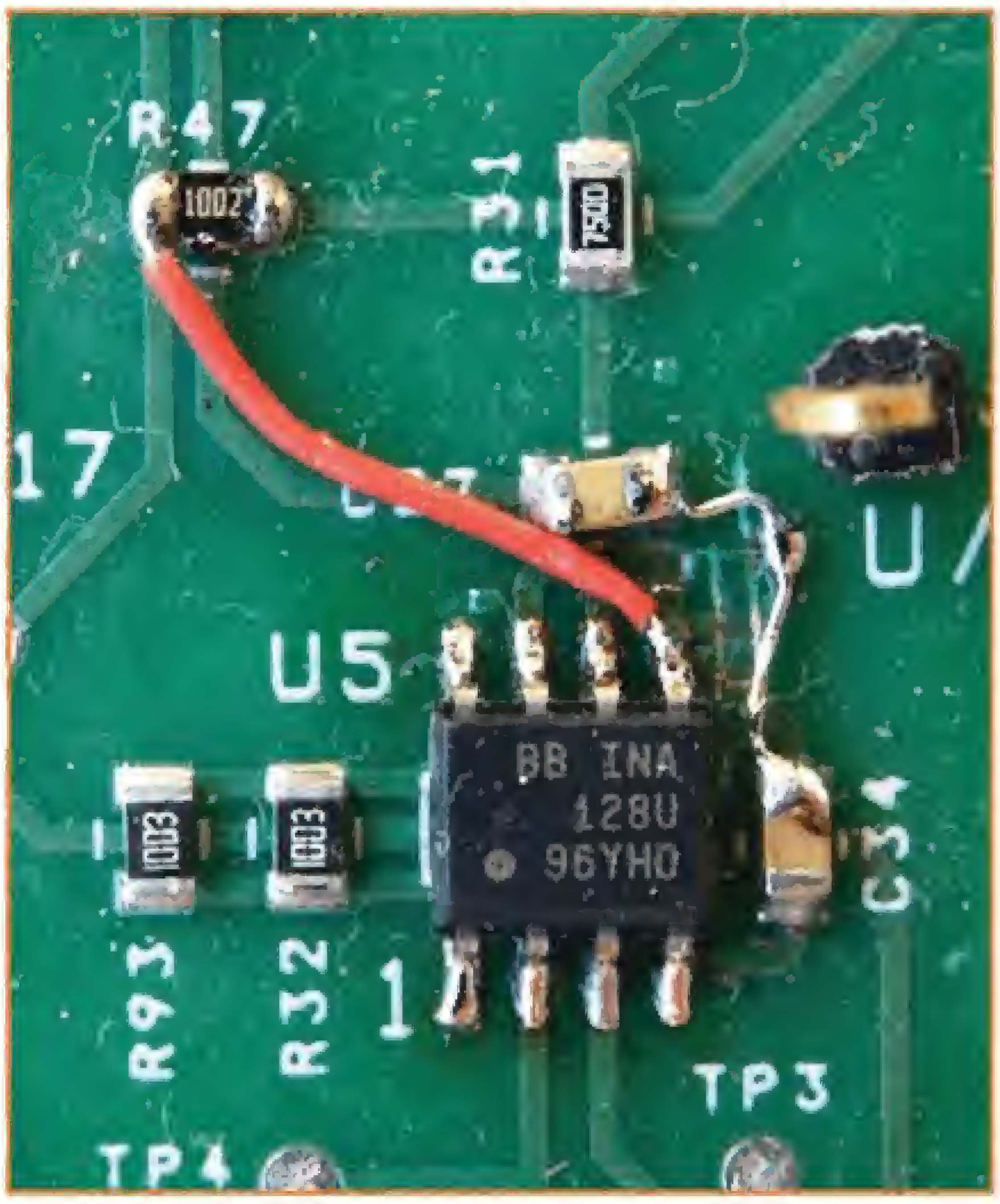


Figure 3.
Don't try to unsolder pin 5 on U5—you risk disaster! Use a scalpel to isolate it, then re-make the connections using wire.

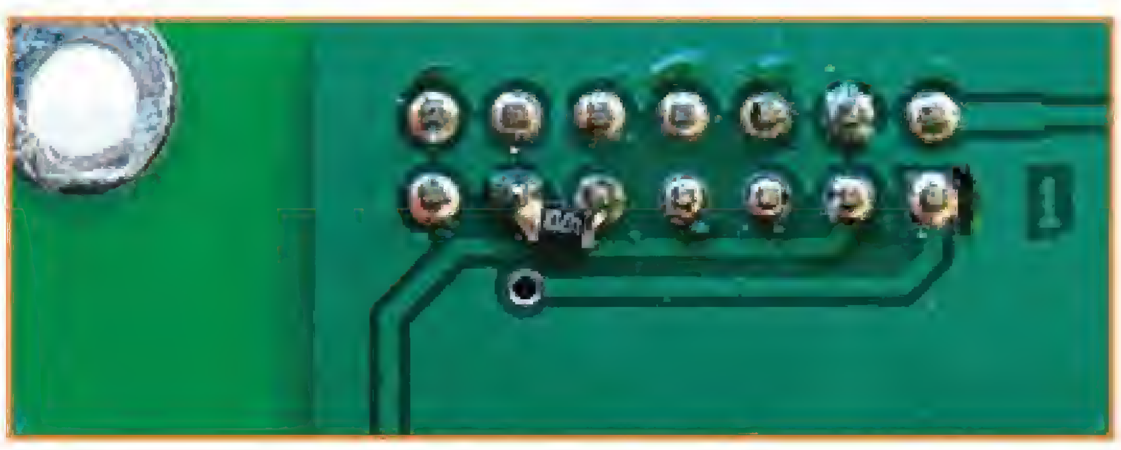


Figure 4.
This resistor tells the firmware that the circuit has been modified as shown in Figures 1, 2 and 3.

New programs [4]

The version 3.0.0 firmware (*LCR3A_firmware_V300.hex*) supports the new input offset voltage compensation circuit if, and only if, a resistance to ground is detected on pin 9 of J17

This results in a re-arrangement in the menus in the *AU2011* program, which also updates to version 3.0.0:

the original *Input_offset adjustment...* menu is replaced by two menus *Input_offset_U adjustment...* and *Input_offset_I adjustment...*, the first with the input shorted, the second with it open circuit.

For users who do not wish (or are unable) to modify the hardware, the solution consists in limiting the overall gain in range 1. Hand-in-hand with this, similar limiting can be performed in range 8, if there is too much interference in the surroundings, preventing the *TRIM - OPEN-CIRCUIT* compensation being performed correctly.

The gain limiting solution is valid whether or not the device has been modified.
The *initial* maximum value will be 5 for an unmodified device and 15 (step F) for a modified device. Naturally, the value modifications are stored in the device's memory, just like the other settings/options in the Preferences window.
It follows that we also have to add two menus in standalone mode, and modify the Preferences window in PC mode when the *access to adjust. menus* option is checked (**Figures 5 & 6**).

Other modifications have been made to improve convenience in use, like the appearance of a "Port / Close the Port" menu, handy if you originally chose the wrong port; all you then have to do is select the right port and click the 'Open COM' button in the main window.

Measuring high impedances

The 'voltage' and 'current' signals are amplified with no low-frequency filtering prior to sampling. It is only after digitizing, by performing the measurement over a whole number of AC powerline cycles and taking the average of several measurements, that it is possible to reduce the influence of the stray signals picked up by the measuring device.
This means it is possible for the signal applied to the analog/digital converter (ADC) to briefly exceed this ADC's input range and invalidate the measurement.

Consequently, particular care must be taken when measuring high impedances, when the LCR meter is set to range 7 and above all 8; this is the case during *TRIM – OPEN-CIRCUIT*.

- Put the electronics in an earthed metal case (iron is preferable to aluminum at low frequencies). Take care if you are using the LCR Meter in standalone mode with a USB supply, or in PC mode with a laptop: in these cases, there is no earth connection and you'll need to make one. The power plug on a USB PSU doesn't have a pin to connect the LCR Meter case to the AC powerline protective earth; and a laptop running on its battery isn't earthed either.

- Minimize the length of the measuring cables, and keep away from power cords. To protect the device and the measuring leads from radiated fields, place a grounded metal plate (preferably iron) of adequate size between them and any powerline wiring.

- If there is still interference, reduce the gain of the measuring chain to range 8 (*Max DACIndex I* in standalone mode).

To conclude, let's just note that experience has shown that in practice, the 4-BNC measuring unit solution (TONGHUI TH26001A or HAMEG HZ181) is very much preferable to the Kelvin clip.

(130307)

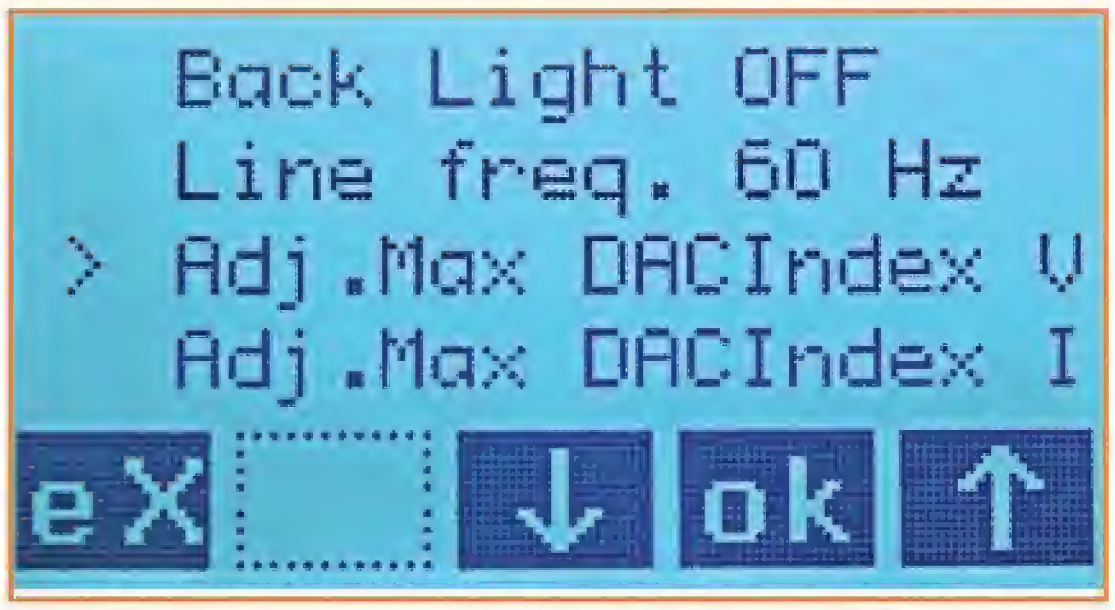


Figure 5.
Two new menus in standalone mode.

Internet Links

[1] 500 ppm LCR Meter
Part 1, Elektor no. 417, March 2013
www.elektor.com/110758
Part 2, Elektor no. 418, April 2013
www.elektor.com/130022
Part 3, Elektor no. 419, May 2013
www.elektor.com/130093

[2] www.elektor.fr/forumLCR (in French)

[3] www.elektor.com/forum/elektor-forums/fields-of-interest/test-measurement.1543743.lynkx

[4] www.elektor.com/130093

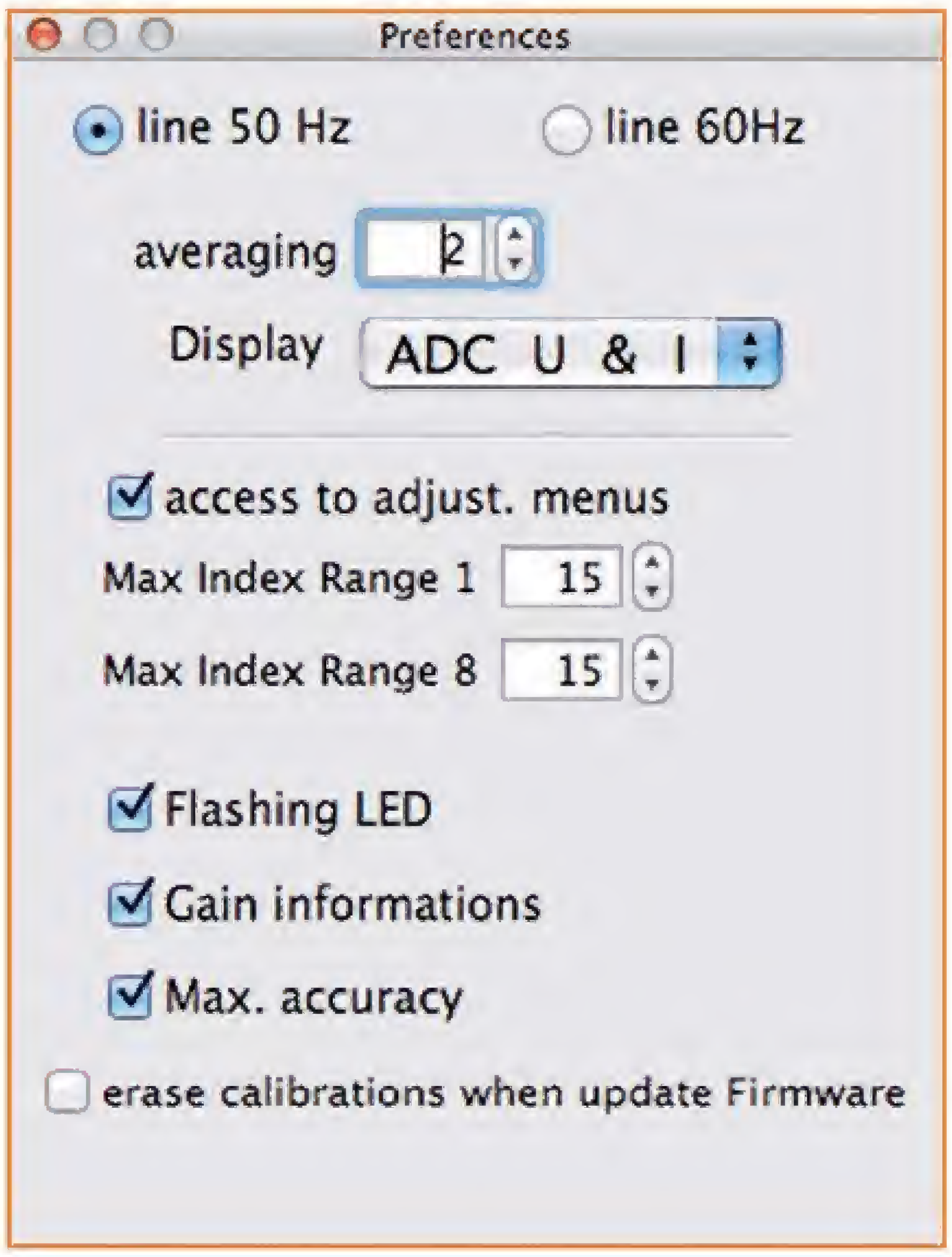


Figure 6.
The 'access to adjust. menus' option under the Preferences in PC mode.

Multichannel Temperature Logger

By **Ihab F. Riad** (Physics Dept., University of Kartoum, Sudan) (a hot place)

This project allows you to log up to six temperature readings over a period of time, complete with time stamps, all written to a .csv file stored on an SD card for processing on a PC. Local control is also available in the form of an LCD and a keypad.

Features

- Max. six DS18S20 1-Wire temperature sensors
- PIC18F4520 based
- 1 second minimum logging interval
- Writes time-stamped .csv data on SD/MMC card
- Local control with LCD and keypad
- On-board RTC

The main components found in this project are the DS18S20 digital temperature sensor, the RTC-DS1338 Real Time Clock, and a microcontroller type PIC18F4520. Due to the ±0.5 °C temperature resolution of the sensors and a minimum logging time of 1 second, this logger is most suitable for environmental monitoring, like your local temperature at six fixed height intervals above the ground.

The sensors

First off, you can use a maximum of six DS18S20 temperature sensors to capture an equal number of temperatures at remote locations. If your application requires just two or three sensors, that's fine too.

Looking at the schematic in **Figure 1**, the remote sensors are connected to 3-pin connectors K3 through K8 using the 1-Wire system (which actually involves three wires). As opposed to some previous projects like our Thermo-Snake [1], here the DS18S20 is used in standard 1-Wire mode rather than in 'parasite power' mode. Rather than being connected to a common line or 'bus', each DS18S20 sensor has its own PIC port line RA0–RA5 and associated resistor network (R22/R28 and so on) connected to its DQ (data in/out) line. The DS18S20 being a 1-Wire component, each device ever produced by Dallas Semiconductors has a unique 64-bit identifier stored in ROM. The device sends messages using the format illustrated in **Figure 2**.

Into the schematic

Returning to Figure 1, RTC chip IC1 supplies the time stamps for the logged data to the micro, using I²C lines SDA and SCL. The DS1338 has its

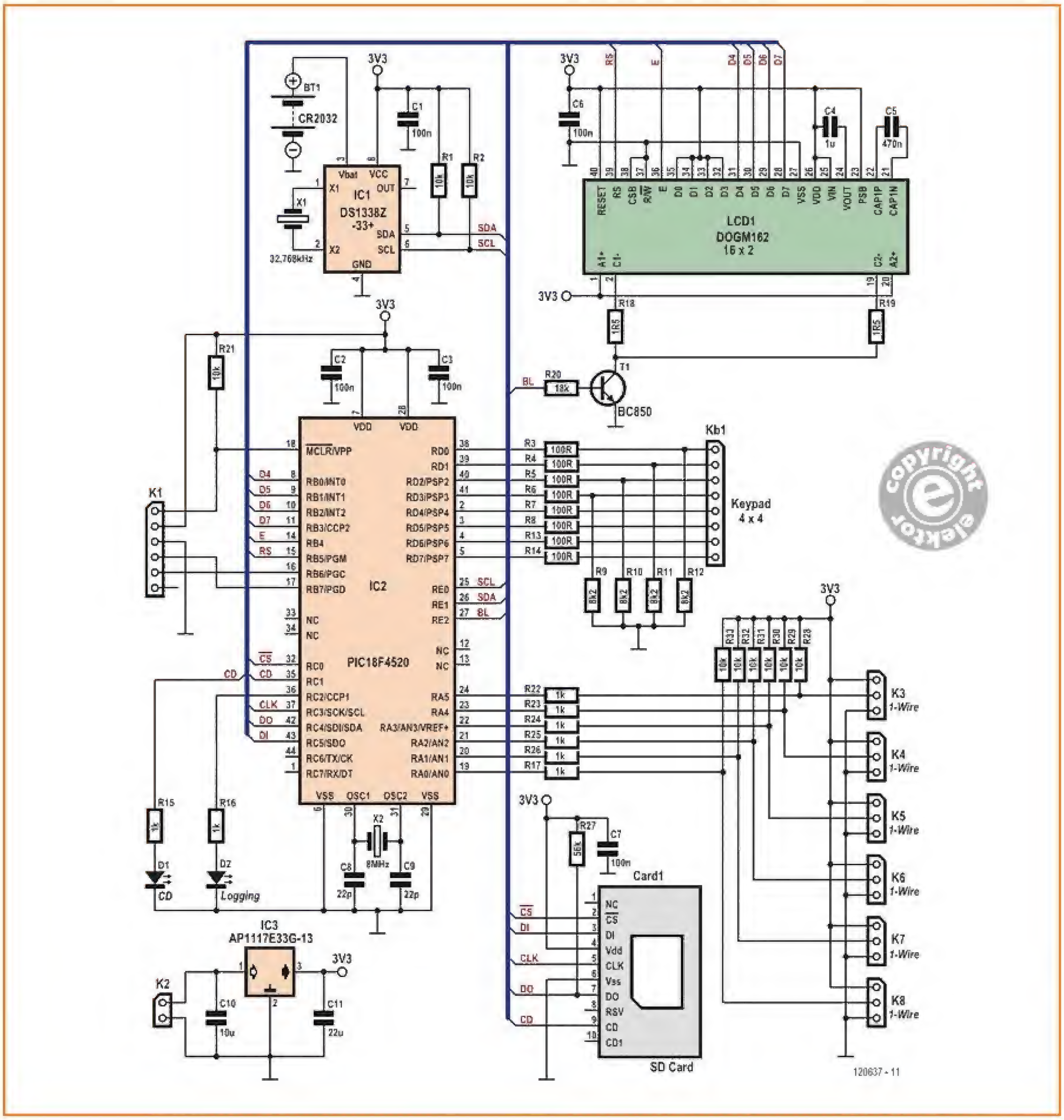


Figure 1.
Schematic of the Multichannel Temperature Logger. Besides yours, some intelligence resides in the PIC18F4520.

traditional 32.768-kHz watch crystal, and operates off a 3.0-volt button cell, BT1, or 3.3 V when the board is powered.

A commercial 4x4 matrixed, numeric keypad on microcontroller port lines RD0–RD7 and connector Kb1 is used for setting the time/date and the logging interval. The keypad is also used to start and stop the logging. A 2-line, 16-character (2x16) LCD type DOGM162 (what's in a name?) is used to display the current time and date, as well as the instantaneous temperature from one of the sensors. The LCD's backlight (BL) function is controlled with T1 responding to control levels issued by the PIC micro on port line RE2. An SD/MMC card on the 'Card1' connector holds the logged data. All read/write access and control of the SD card is via five lines on microcontroller port RC. The card can be removed and the file on it read on a PC for processing by your favorite statistics or graph rendering program capable of processing .csv files. Lots of nicely colored graphs in particular work wonders on CFOs, CEOs, CCOs, CYOs, CXYZOs and other non-electronics initiated persons in the audience.

The internal fuse settings enable the microcontroller to be clocked by an external 8-MHz quartz crystal, X2. With the internal PLL enabled, the PIC's actual CLK is 32 MHz. The crystal is flanked by the

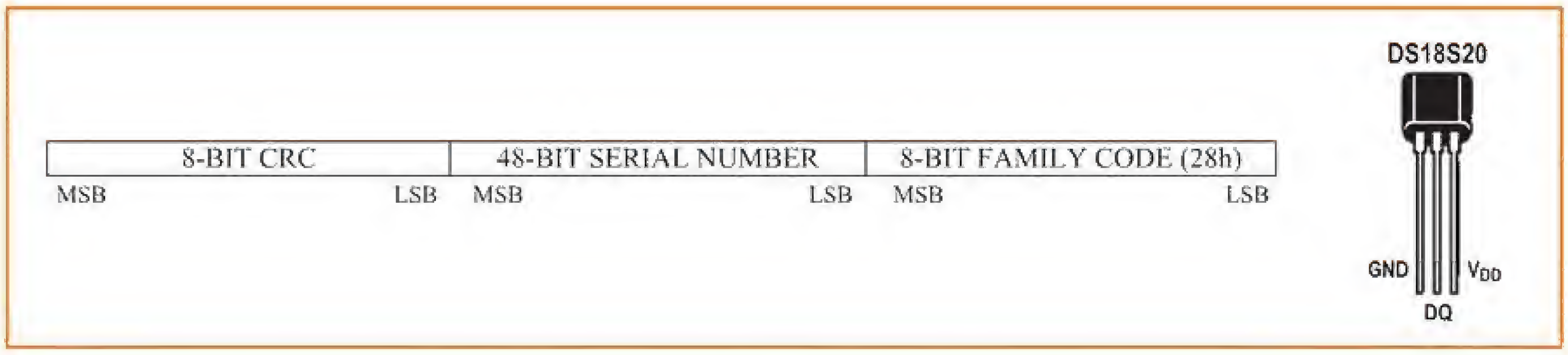


Figure 2.
1-Wire message format and pinout of Dallas' DS18S20 1-Wire temperature sensor.

customary pair of 22-pF ceramic load capacitors. A word or two about these apparently paltry little parts. Get these wrong and funny things may happen. Like C or C++ experts and other programming gurus fitting "yeah-well-something" parts in the "elector" circuit and subsequently spending hours on debugging the code, meanwhile creating long forum threads all across California right up to MIT Boston and across the ocean to Limbricht, all because the micro is running at a speed that's wildly different from what the designer planned. The upshot: get the xtal load capacitors wrong and your PIC oscillator will not work.
Back to digitalism, connector K1 is the gateway to the Microchip PICKit.
Two LEDs are provided: D2 to show logging activity, and D1 to show card detection.
The 3.3-V supply voltage for the entire circuit is furnished by a low-drop regulator, IC3. The maximum input voltage will be about 18 V (but don't push it), the minimum, about 4.6 V. Four AA 1.5-volt dry batteries will last a long time.

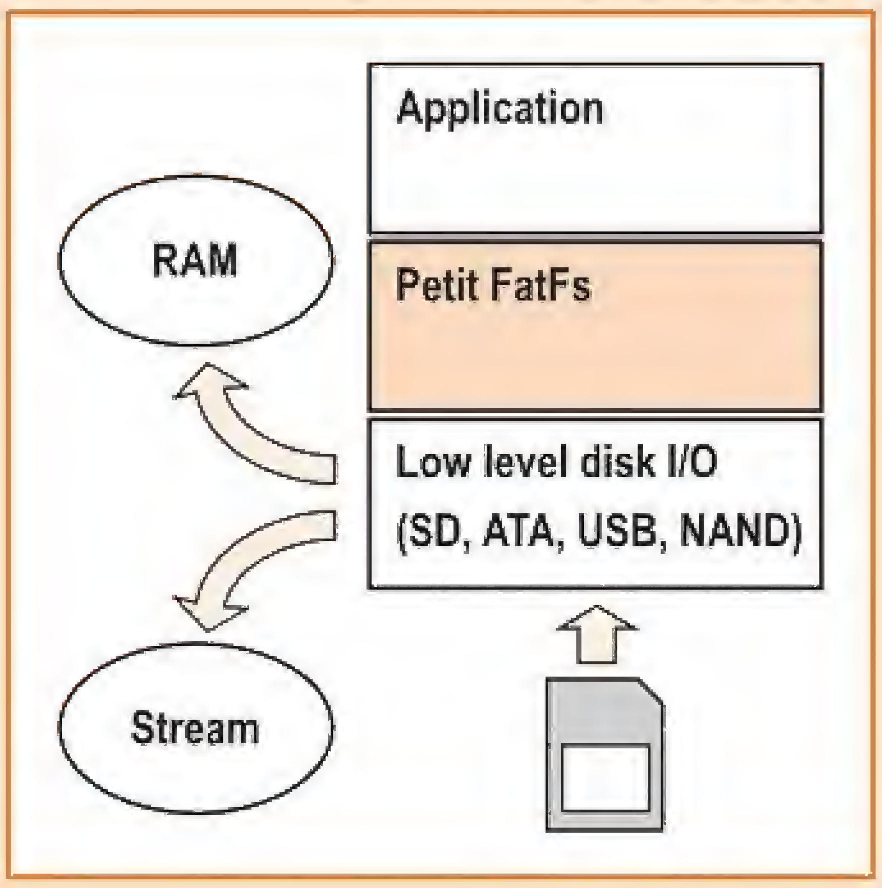
Applications and how it was developed

The first unit was built by the author to monitor the temperature variation at different points in a concrete slab just after the mixture was molded, and during hardening. That was for an M.Eng. student. The unit was primitive at that time with no more than the keypad and the LCD. The logging was done manually every few hours for a couple of days.
One member at www.elektor-labs.com suggested putting four sensors on a stick at 1-foot (30-cm) intervals above the ground and two sensors in the ground. This will give you a good idea of the temperatures in your garden.
The original code was written using PIC MIKROC from Mikroelektronika. Testing and debugging was carried out with the help of an Easypic6 development board from Mikroelektronika, their RTC2 module, and their MMC/SD board. The author's

Petit FAT Fs

PetitFAT File System (Petit FATFs) is written in compliance with ANSI C, and completely separated from the disk I/O layer. It can be incorporated into tiny microcontrollers with a small memory even if the RAM size is less than sector size.
Petit FatFs features include

- very small RAM consumption (44 bytes work area + certain stack);
- very small code size (2-4 Kbytes);
- supports FAT32;
- single volume and single file;
- file write function with some restrictions.



In terms of the Application Interface, Petit FATfs provides the following functions: pf_mount (mount/unmount a Volume); pf_open (open a file); pf_read (read file); pf_write (write file); pf_lseek (Move read/write pointer); pf_opendir (open a directory); and pf_readdir (read a directory item).
PetitFat Fs is completely separated from the disk I/O layer, hence it requires certain lower-layer functions to read the physical disk. The low level disk I/O module is not a part of Petit FatFs module and it must be provided by user. The sample drivers called disk_initialize, disk_readp (partial) and disk_writep (partial) are also available in the resources [4].

original test setup is pictured in **Figure 3**. While processing the project for publication here, new software was developed by Elektor Labs using their Microchip MPLAB X environment and C18 compiler.
The SD card is read, and written to using Petit FatFs, a sub-set of the FatFs module for 8-bit microcontrollers, see **inset**.
The 1-Wire protocol got implemented with the help of a C18 library at [2].
All PIC source code files for the project have been packaged into a .zip archive file to be found at [3] for free downloading.

The log file

The temperature logger expects a file called TempLog.csv on the SD card. The logger is unable to create a new file or adjust the size of a file. At the start of a new log session this file is opened and overwritten starting from the first record. Consequently, some measurement values remain in place if the new session contains fewer samples than the previous one. It is therefore recommended to use your PC to write a new, blank file to the SD card prior to a new logging operation. This empty file is included with the free software download from the Elektor website [3]—but can also be created from scratch, see [4]. Quick & dirty, to create a new log file, type:

```
fsutil file
createnew\<[driveletter>:<file size in
bytes>
```

The default size of our file is 5 MB, but tailor the size to your liking.
In terms of measured values appearing on the LCD, T(emp.)0 is the sensor wired to K8 (nearest the display), T1 to K7, and so on, up to T7 on K3 at the lower side of the PCB.
A new line/record is written into the TempLog.csv file for each new measurement. Columns 1 and 2 contain date and time respectively. In the next columns the measured temperatures appear in order (see above paragraph) of the connected sensors. For example, if only one sensor is connected, its output value appears in the third column, regardless of the connector it is wired to or plugged into.

So You Think You Can

Have a go at changing that 8-MHz existing microcontroller oscillator frequency. Feel free to do so—here's what you're up against in terms of delays that need to be adjusted:

- in file `Globals.c`: adapt `delay_ms`, `delay_us`, `setup_io (SSPADD)`.
- in file `LCD.c`: adapt `XLCDdelay15ms`, `...4ms`, `...100us`, `...500ns`, `XLCDdelay`.
- in file `SW_I2C.c`: all functions.
- in file `Onewire.c`: `ow_reset`, `ow_write_`

Figure 3.
This how the project got developed at the author's home using a Mikroelektronika EasyPIC6 development system and some add-ons.

Key Functions

The keypad is basically a DTMF (telephone) type with numbers 0–9, letters A–D, the hash sign (#), and an asterisk (*). The key functions are summarized below. Keys presses do not produce DTMF sounds.

A: Adjust logging interval
  0-9: Change number (advances automatically)
D: Exit the setting menu
Note that after every unit (hour, minutes, seconds), that unit is updated only—not the rest.

B: Start/Stop logging (LED indicates logging in progress)

C: Set Clock
  0-9: Change number (advances automatically)
D: Exit the setting menu
Note that after every unit (hour, minutes, seconds), that unit is updated only—not the rest.

0 – 5: Selects sensor to be displayed on LCD

Display Codes:															
Startup:															
C	L	K	:		wd	wd		d	d		mo	mo		y	y
R	E/U	S/N						h	h		mi	mi		s	s
Home (x = sensor number):															
T	x	:			wd	wd		d	d		mo	mo		y	y
(-)	T	T	.	T				h	h		mi	mi		s	s
Set time:															
S	e	T			wd	wd		d	d		mo	mo		y	y
t	i	m	e					h	H		mi	mi		s	s
Adjust log:															
S	e	t		l	o	g		i	n	t	e	r	v	a	l
h	:	h	h		m	:	m	m		s	:	s	s		

`byte`, `ow_read_byte`, `ow_read_bit`, `ow_get_temperature`.
- in file `mmc.c`: `disk_initialize`, `init_spi`.

All done? Then post your results to the community at www.elektor-labs.com.

Construction

The circuit board designed by Elektor Labs for the project is shown in **Figure 4**, along with the parts list. The board's overall shape and dimensions are governed by the LCD and the keyboard installed on top of it. Etchers@home: the PCB .pdf files are at [3]. Apart from a good number of 0.1'' pitch pinheaders, a button cell holder and 0.1'' pitch socket strips with turned pins, the board contains mostly SMD parts. Of these, the PIC microcontroller will be the most bothersome to fit, but work calmly and accurately and it can be done. Methods of hand soldering these multi-legged parts successfully have been described many times.

The keypad is mounted on four 15-20 mm standoffs to clear the clip on the battery holder. Finally, the LCD is a fragile device and must be handled and mounted with the utmost care.

(120637)

COMPONENT LIST

Resistors

(SMD 0805)
R1,R2,R21,R28,R29,R30,R31,R32,R33 = 10kΩ 5% 125mW
R3,R4,R5,R6,R7,R8,R13,R14 = 100Ω 5% 125mW
R9,R10,R11,R12 = 8.2kΩ 5% 125mW
R15,R16,R17,R22,R23,R24,R25,R26 = 1kΩ 5% 125mW
R18,R19 = 1.5Ω 5% 100mW
R20 = 18kΩ 5% 125mW
R27 = 56kΩ 5% 125mW

Capacitors

(SMD 0805)
C1,C2,C3,C6,C7 = 100nF 50V 20%
C4 = 1µF 16V
C5 = 470nF 25V
C8,C9 = 22pF 50V 5%
C10 = 10µF
C11 = 22µF 10V

Semiconductors

D1,D2 = LED, 3mm, low current
T1 = BC850, NPN 45V transistor, SOT-23
IC1 = DS1338, real time clock, SOIC8
IC2 = PIC18F4520-I/PT, 8-bit MCU, programmed, Elektor Store # 120637-41
IC3 = AP1117E33G, LDO regulator, 3.3V, SOT223
IC4,IC5,IC6,IC7,IC8,IC9 = DS18S20, 1-Wire temperature sensor, TO92 (not on board)

Miscellaneous

Kb1 = MCAK1604NBWB, keypad, 4x4 array, Multicomp
X1 = 32.768kHz quartz crystal, 12.5pF load, 20ppm, 4.1x1.5mm, Abracon ABS09-32.768KHZ-T
X2 = 8MHz quartz crystal, 18pF load, 20ppm, 5x3.2mm, Abracon ABM3-8.000MHZ-D2Y-T
Card1 = uSD (micro SD) connector, Hirose DM3AT-SF-PEJM5(40)
BT1 = CR2032, with PCB mount holder
LCD1 = DOGM162W-A 2x16 character LCD
Backlight EA LED55x31-G
K1 = 6-pin pinheader, right angled, 0.1'' pitch
K2 = 2-pin pinheader, right angled, 0.1'' pitch
K3,K4,K5,K6,K7,K8 = 3-pin pinheader or socket, right angled, 0.1'' pitch
Socket strip 0.1'' pitch, turned pins, for mounting LCD and keypad
PCB # 120637

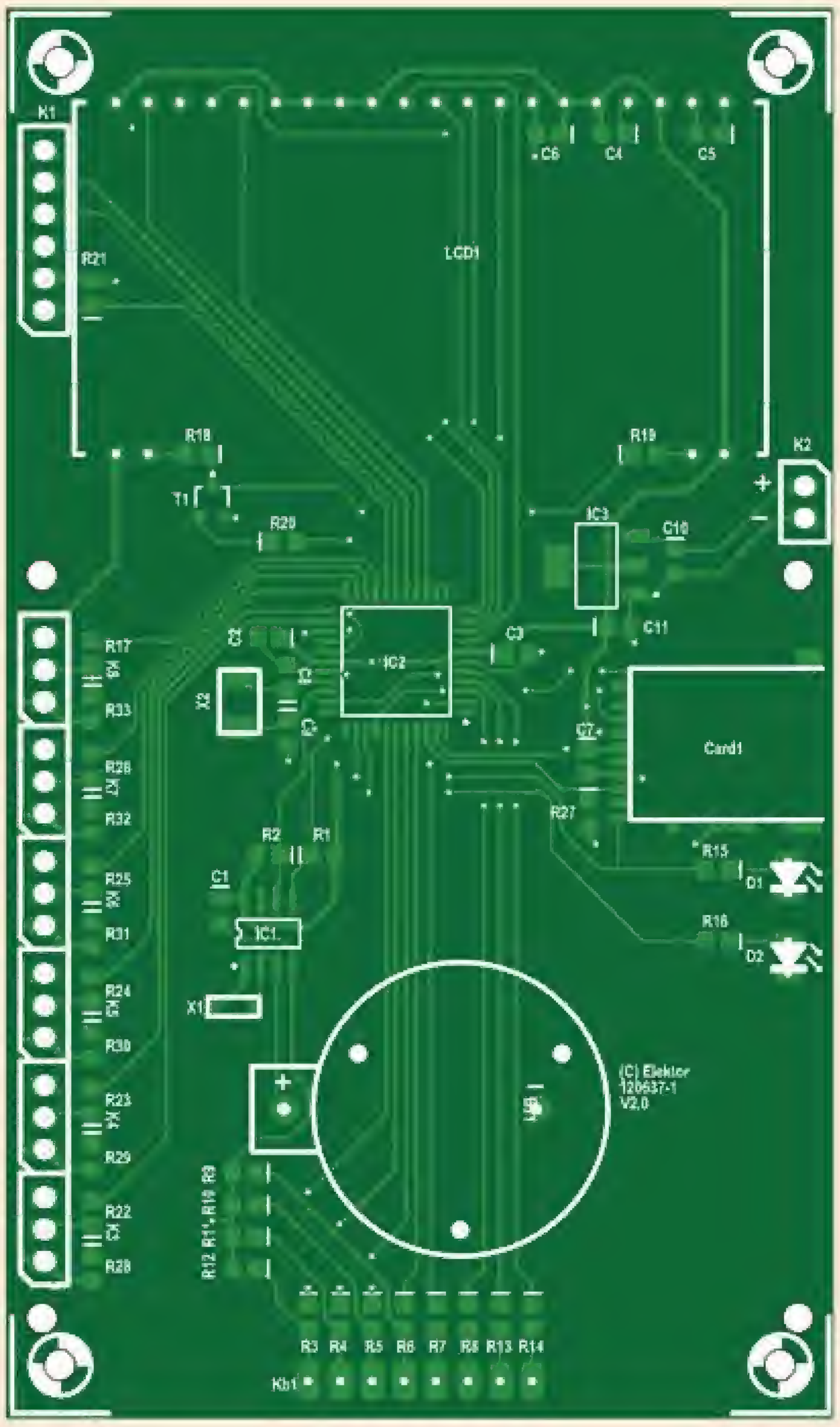


Figure 4.
The printed circuit board designed for the project contains chiefly SMD parts.

Internet References

[1] Thermo-Snake, Elektor June 2008, www.elektor.com/070122

[2] 1-Wire protocol: http://psychoul.com/electronics/1-wire-onewire-c18-library-2

[3] www.elektor.com/120637

[4] Create a file of size xx: http://windowsitpro.com/systems-management/how-can-i-create-file-certain-size-windows-xp-and-later

[5] PetitFAT Fs: http://elm-chan.org/fsw/ff/00index_p.html

USB Thermometer

A simple method to read data from the USB port

By **Michael Odenwald**
(Germany)

For many years the serial RS232 computer port could be relied upon as a sort of MacGyver universal port. Nowadays PCs are unlikely to be supplied with this 9-pin sub-D connector. When you need to hook up some electronics to a computer, the USB port looks like the only option. The computer however needs a corresponding software driver. Here we describe an elegant patch around this problem.

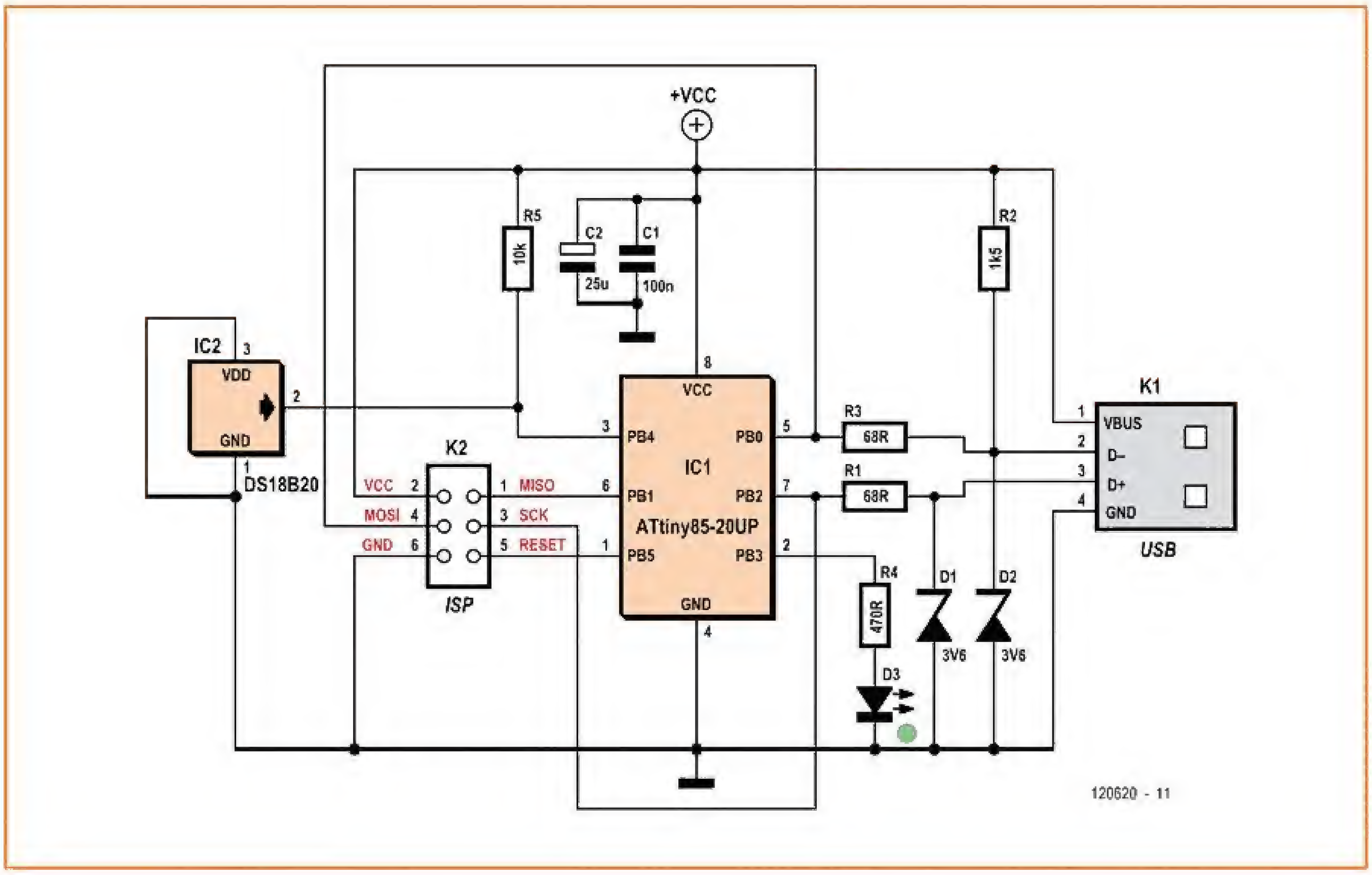


Figure 1.
The USB thermometer circuit is simple.

The development of a software device driver, including adaptation to run in different operating systems, is anything but trivial. Add to this also the niceties of digital signatures for which the device/driver/operating system/application program chain adds additional complexity. The necessary time investment is often unacceptable for small projects. For this reason a virtual COM port is often employed but this brings with it the (configuration) disadvantage and misses out on many of the good USB features.
One USB mode that can be relied on to function and also supports prototype development is the USB-HID (**H**uman **I**nterface **D**evice) class. The USB HID usage allows many more devices than just a mouse and keyboard (see 'The USB HID device class'). The USB standard [1] specifically allows the use of 'other devices' which include all types of actuators and sensors.

Since all of the major computer operating systems already have USB HID drivers there is no reason why you shouldn't use them for your own purposes. You only need to produce a suitable application program to run in your computer's operating system.

Data capture

You can use the USB HID pathway to pass all sorts of external data to a PC. The temperature measurement application is just one example of the process. A small ATtiny microcontroller type ATtiny85-20 (IC1 in the circuit diagram **Figure 1**) is used here to provide the limited degree of 'intelligence' necessary to handle the USB protocol stacks, communications and also to input and format the sensor readings.
All components in the design derive their power directly from 5 V available at the USB socket K1. The microcontroller is clocked at 16.5 MHz from its internal PLL, this gives enough speed to handle USB communications and eliminates an external crystal.
IC2 is the DS18B20 temperature sensor from Dallas semiconductor (now part of Maxim) which uses a 1-wire interface. It operates in so-called Parasitic Power mode [2] where the chip's VDD pin is connected to ground and power is derived from the data signal connection. This method has the benefit of improving accuracy by reducing the effects of self-heating in the chip. The USB data signals are connected via limiting resistors R1 and R3 which restrict current flow from the line drivers in the event of a short circuit. The two 3.6 V zener diodes D1 and D2 ensure that the data signal voltage swing does not exceed the chip's supply voltage.
Resistor R2 is used during USB enumeration and signals to the host (in the PC) that a low speed device (1.5 Mbit/s maximum data rate) is connected.
Capacitors C1 and C2 provide supply decoupling and buffering of the 5 V feed from the USB socket. K2 is the standard 6-way ISP pin header to program the controller. LED D3 indicates an active measurement cycle. With the temperature sensor at maximum resolution this takes around 750 ms.

Firmware

Firmware for the USB thermometer is written in C. The WinAVR development tools [3] are used to compile and burn the program to the microcontroller's flash memory. The USB stack structure is implemented with the help of V-USB software [4]. Functions used to read the temperature sensor are taken from a library by Martin Thomas [5]. After the hardware is initialized the USB software stacks and USB enumeration process is executed. The firmware then switches to internal operational mode where a state machine is implemented. This Finite-State Machine consists of: USB protocol, read the sensor and then wait. The states are sequentially cycled with a predefined delay period.
Each complete measurement cycle takes 10 s. Requests from the host within this period will return the same value, only after each cycle has

The USB-HID device class

The HID (**H**uman **I**nterface **D**evice) device class is a partial definition of the USB standards describing devices which provide input from the user. Typical examples would be a keyboard, mouse or joystick. In addition to these normal types of input devices the USB standard also caters for 'special systems' which can take the form of sensors, measuring equipment or even telephones or headsets. The use of readers, games items and promotional products are also anticipated. USB-HIDs have the benefit that the corresponding system driver is already contained in the computer's operating system (at least for Windows, Linux and OS X) and is therefore automatically loaded, without any input from the user, whenever a new USB HID is connected.
The disadvantages of HIDs should also not be overlooked: The data transmission rate is not particularly high and they have a limited number of USB Endpoints, restricting the amount of data which can be transferred.

The most important element in the firmware is the USB HID Descriptor:

```
/*
 * USB HID report descriptor
 */
PROGMEM char usbHidReportDescriptor[33] =
{
    0x06, 0x00, 0xff,       // USAGE_PAGE (Generic Desktop)
    0x09, 0x01,             // USAGE (Vendor Usage 1)
    0xa1, 0x01,             // COLLECTION (Application)
    0x15, 0x00,             //   LOGICAL_MINIMUM (0)
    0x26, 0xff, 0x00,       //   LOGICAL_MAXIMUM (255)
    0x75, 0x08,             //   REPORT_SIZE (8)
    0x85, 0x0a,             //   REPORT_ID (10)
    0x95, 0x04,             //   REPORT_COUNT (4)
    0x09, 0x00,             //   USAGE (Undefined)
    0xb2, 0x02, 0x01,       //   FEATURE (Data,Var,Abs,Buf)
    0x85, 0x14,             //   REPORT_ID (20)
    0x95, 0x0a,             //   REPORT_COUNT (10)
    0x09, 0x00,             //   USAGE (Undefined)
    0xb2, 0x02, 0x01,       //   FEATURE (Data,Var,Abs,Buf)
    0xc0                    // END_COLLECTION
};
```

COMPONENT LIST

Resistors
R1,R3 = 68Ω
R2 = 1.5kΩ
R4 = 470Ω
R5 = 10kΩ

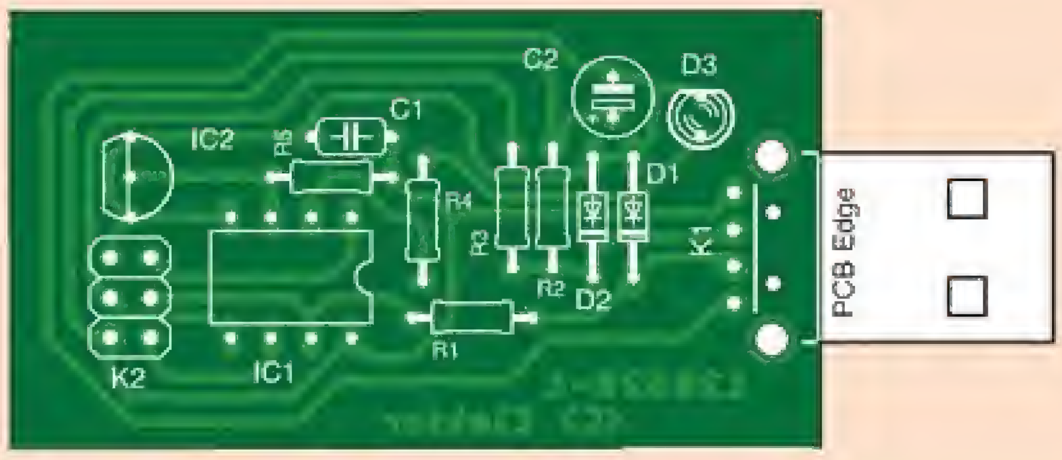


Figure 2.
The PCB component placement.

Capacitors
C1 = 100nF ceramic, 5mm pitch
C2 = 25µF 16V, electrolytic, 2.5mm pitch

Semiconductors
IC1 = ATtiny85-20UP, 8-pin DIL, programmed, Elektor # 120620-41 [6]
IC2 = DS18B20, 3-pin TO92 case
D1,D2 = ZF3.6, zener diode, 0.5W
D3 = LED, green, 5mm

Miscellaneous
K1 = USB socket, Type A, PCB mount
K2 = 6-pin (2x3) pinheader, 0.1'' pitch
PCB # 120620-1 [6]

elapsed will the latest measurement be available. The measurement process is controlled completely by the microcontroller and not by the PC. The measurement interval helps reduce sensor self-heating effects.
The Descriptor consists of 33 bytes. It defines the possible report IDs (10 and 20), the application program uses these to communicate with the thermometer. The reports are implemented as so-called 'Feature Reports' with different length information blocks (4 and 10 byte). A feature report allows values to be read from or written

The central class for the USB thermometer is then:

```
namespace WindowsApp
{
    /// <summary>
    /// Implementation of the usbDevice with service methods
    /// based on the class usbGenericHidCommunication
    /// </summary>
    class usbDevice : usbGenericHidCommunication
    {
        private int tval;

        /// <summary>
        /// Class constructor - place any initialization here
        /// </summary>
        /// <param name="vid"></param>
        /// <param name="pid"></param>
        public usbDevice(int vid, int pid)
            : base(vid, pid)
        {
        }

        /// <summary>
        /// USB HiD Temperature Module Method GetTemeratur()
        /// </summary>
        public int GetTemperatur()
```

to the USB HID system; in this application they are only read.
Report ID 10 is used to request the temperature and returns four bytes. Report ID 20 queries an ident string consisting of 10 bytes giving date information (yyyy-mm-dd).

Build then drive

The circuit layout is not at all critical and the use of non-SMD components with the PCB designed for this project (see **Figure 2**) makes construction really easy. The PCB layout files are also available for free download from the Elektor website for this article [6]. **Figure 3** shows the fully populated prototype. The design does not need any set-up or calibration procedure.

Once all the components have been fitted and you have double checked your handiwork the firmware can be programmed into the microcontroller. Seek out the firmware which is freely available as both a source code file and a directly programmable hex file from [6]. Plug in your AVR-ISP programmer to connector K2 to program the device. It is important to check that the 'divide by eight' clock divider option is deactivated and that the correct internal clock is selected. The settings are correct when the 'low fuse' has the value 0xE1

Figure 3.
The USB thermometer prototype.

```
        {
            // Declare an input buffer
            Byte[] inputBuffer = new Byte[5]; // we expect 5 byte; 1 x ReportID and 4 Byte temperature

            inputBuffer[0] = 10; // Read ReportID 10

            // Perform the Read Command
            bool success;
            success = getFeatureReport(inputBuffer);

            if (success == false)
            {
                Debug.WriteLine("Error during getFeatureReport");
                return tval;  // Error during USB HiD_GetFeature Request so return the old value
            }

            tval  = inputBuffer[1] << 24;
            tval |= inputBuffer[2] << 16;
            tval |= inputBuffer[3] << 8;
            tval |= inputBuffer[4];

            return tval; // Return the new value
        }
    }
}
```

and the 'high fuse' has the value 0xDD.

Once the microcontroller has been programmed the circuit can be hooked up to the PC using a USB cable. The operating system will detect that a new HID device has been plugged in and will automatically install the HID system driver; it's that simple! It doesn't matter is you are using a 32 or 64 bit version of Windows, OS X or Linux, the HID driver is always available, always digitally signed and will always install without any further input from the user. A couple of seconds after plug-in the unit is ready to go.

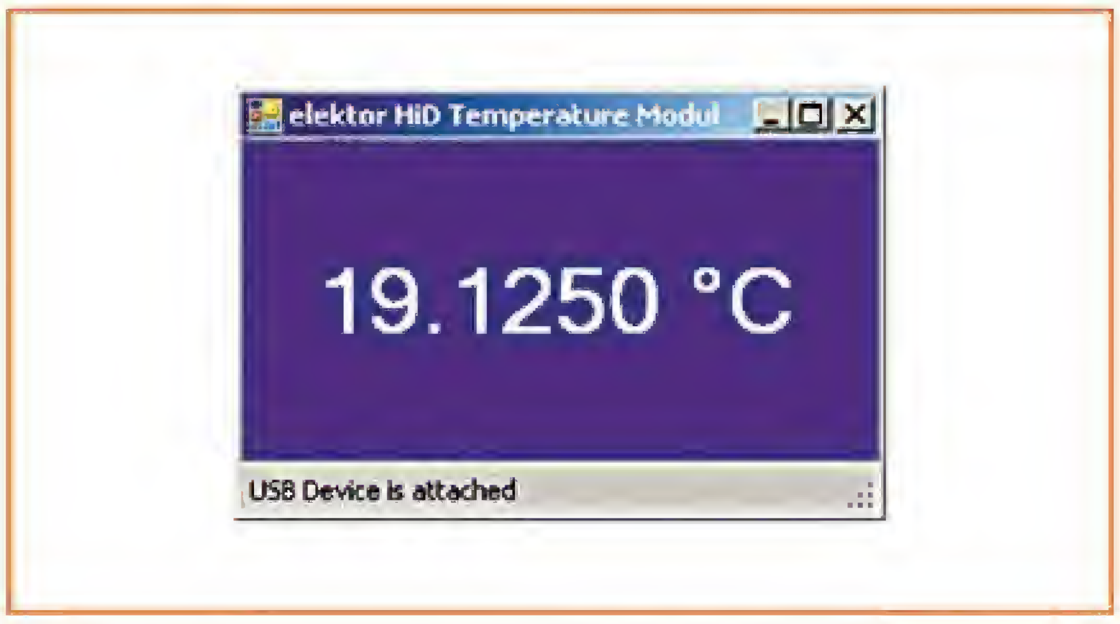


Figure 4.
Temperature displayed by the application program running in Windows.

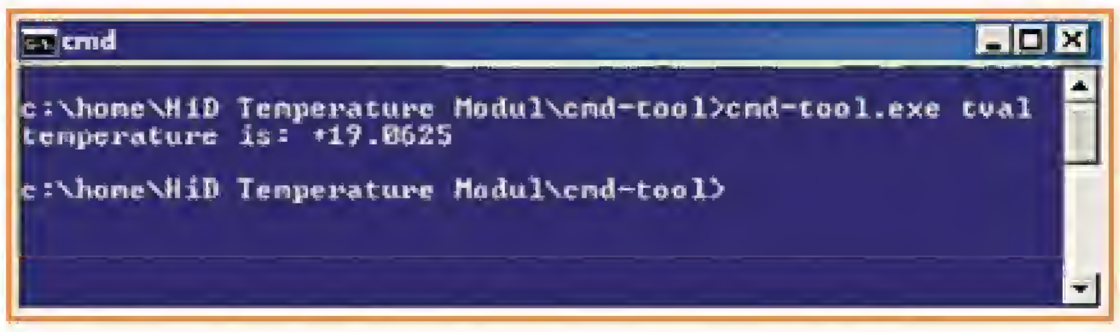


Figure 5.
Temperature display using the command line tool.

The host software

The (Windows) application program which inputs and displays the USB thermometer readings is written in C#. It demonstrates how communication with a generic HID device is performed. The host software uses the usbGenericHIDDevice functions in the library [7], which interacts with the Windows API functions. The software is compiled and run using the express version of Visual Studio 2010 [8]. The base class usbGenericHidCommunication is key here, from which a class for our own HID device must be derived. In this class the 'method' to be executed is implemented.

The HID device is identified and called with the parameters Vendor ID = 0x0C7D and Product ID = 0x0011. The method GetTemperatur() reads and returns the temperature value. The temperature range of the sensor spans -55 to +125 °C. The sensor sends measurements formatted according to 'signed longint' convention giving values in the range −550,000 to +1,250,000. The software divides these values by 10,000 giving a temperature value with a resolution of 12 bits or 0.0625 °C. This level of resolution is not strictly necessary because the sensor itself has an accuracy of only 0.5 °C. The screenshot in **Figure 4** shows how the temperature is displayed in Windows.

The Windows compliant source code together with the necessary library is available to download for free [6]. In addition to this Windows based program there is a simpler command line tool that returns the temperature reading as text (**Figure 5**). The corresponding source code, compiled program and tool for Linux are also available from [6].

And so it goes...

The circuit and corresponding software for the USB thermometer demonstrates a simple and practical method to connect your own HID device to a PC. Both can be relatively easily adapted to the needs of your own particular application, giving an (almost) hassle-free pathway to pass data to a PC providing you do not have too much data to send and can accept a relatively modest data rate.

(120620)

Internet Links

[1] www.usb.org/developers/hidpage/

[2] http://datasheets.maximintegrated.com/en/ds/DS18B20.pdf

[3] http://winavr.sourceforge.net/

[4] www.obdev.at/products/vusb/index.html

[5] www.siwawi.arubi.uni-kl.de/avr_projects/tempsensor/index.html

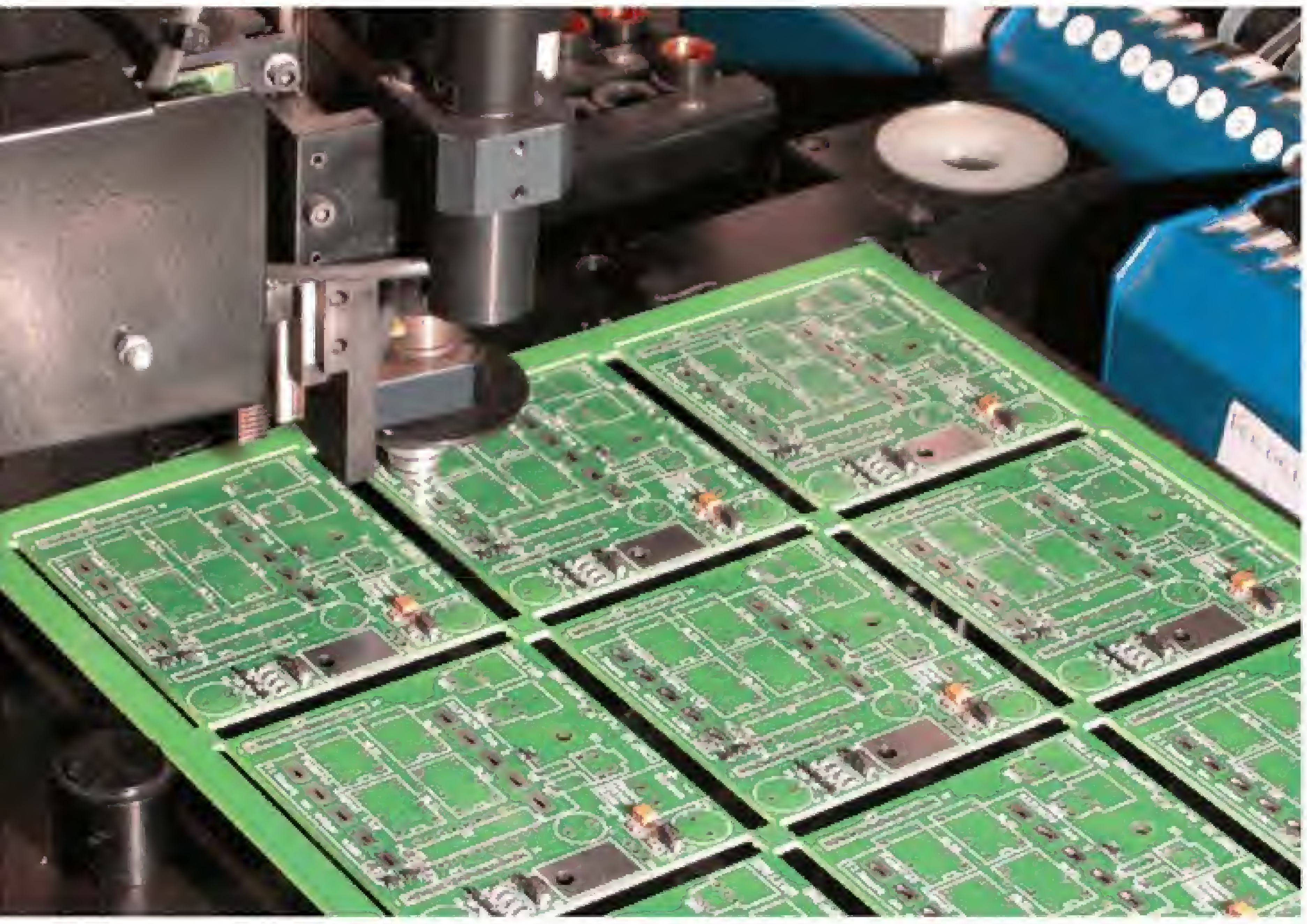
[6] www.elektor.com/120620

[7] www.waitingforfriday.com/index.php/Open_Source_Framework_for_USB_Generic_HID_devices_based_on_the_PIC18F_and_Windows

[8] www.microsoft.com/germany/express/products/windows.aspx

DesignSpark Tips and Tricks

Day #5: generating PCB manufacturing files



By **Neil Gruending** (Canada)

Today we will generate the Gerber files and BOM information for the design we laid out last time. DesignSpark has excellent support for generating these types of files once it's been configured properly.

Gerber Outputs

Gerber files are the file format needed by a printed circuit board (PCB) manufacturer to actually manufacture a design. Think of it as a simple vector image format of the PCB using different size pens (apertures). Gerber files are commonly referred to as plot files because they're used by a photoplotter as part of the circuit board manufacturing process.

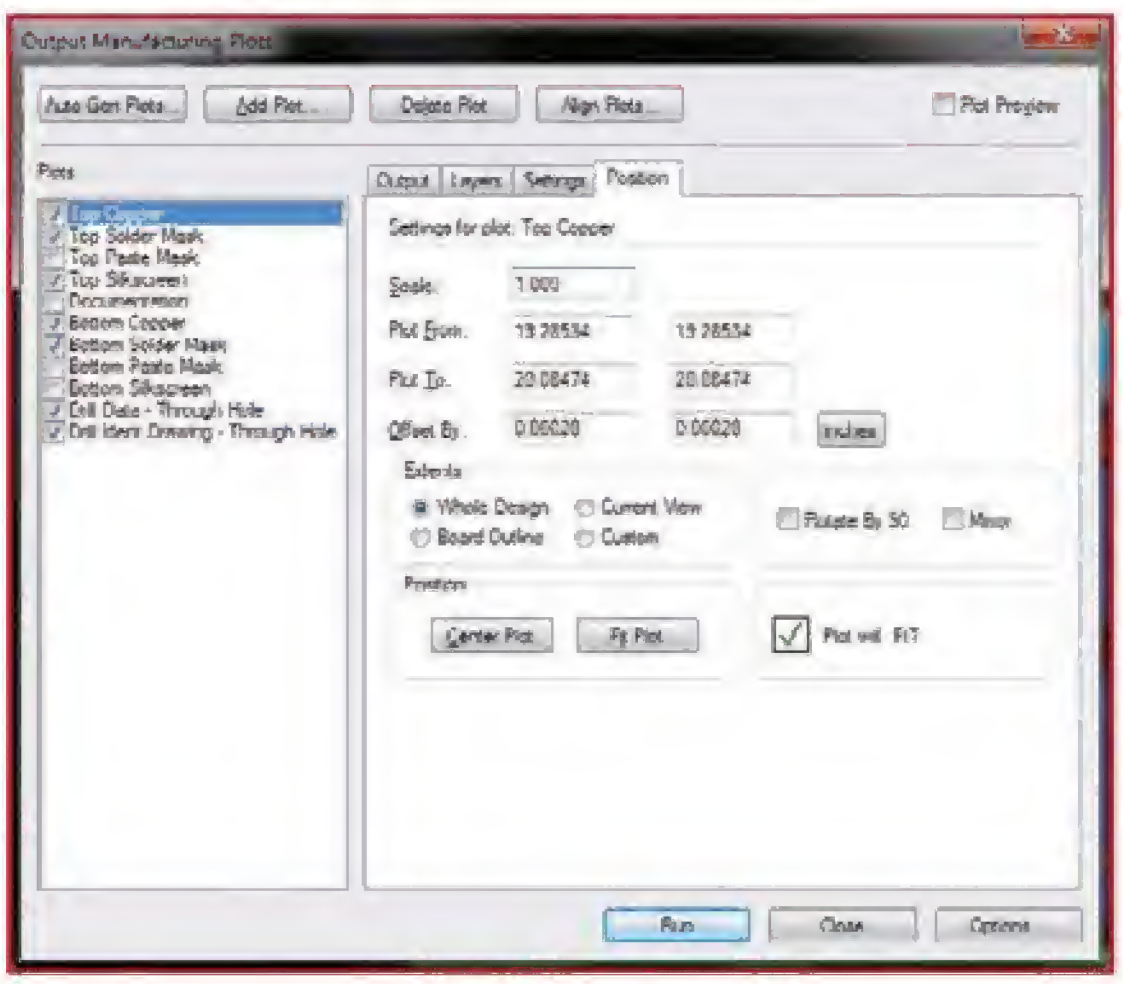


Figure 1.
Data to be included in the Gerber output file.

DesignSpark can generate Gerber files from the "Output Manufacturing Plots" window, which is opened from the Output->Manufacturing Plots menu. In our case we need the Top Copper, Top Solder Mask, Top Silkscreen, Bottom Copper, Bottom Solder Mask, Drill Data and Drill Ident Drawing plots as shown in **Figure 1**.

Clicking on the Options button will open the Options windows where you can modify all of the settings for Gerbers, drill file and PDF outputs. I personally like to use RS-274-X to embed the aperture information into the Gerber files, and to export everything in metric with four decimals of precision as I've illustrated in **Figure 2**.

Clicking the RS-274-X button will adjust the listed options to enable the embedded aperture table and clicking the RS-274-D button will disable all of the output options.

Circuit board manufacturers also need a drill file or NC (numerically controlled) to know where to drill the holes in the circuit board. There are several different output formats but I personally like to use a metric Excellon format with four decimals of precision, as illustrated in **Figure 3**.

Now that we've configured the output devices, we need to add the board outline to all of the generated plots so that the circuit board manufacturer can line up all of the different layers when making the PCB. You do that by clicking on the Layers tab for each plot in the Output Manufacturing Plots window, and double clicking on "[Board Outline]" so that a Y is displayed in the Selected column.

Once the board outline is added to all of the layer plots, click the Run button to generate all of the output files. After the files have been generated, DesignSpark will display a report summary in Notepad, which you should review to make sure that there weren't any errors. All of the Gerber files will have a .GBR extension, and the drill files will have a .DRL extension. I always load the Gerber and drill files into a 3rd party Gerber viewer like ViewMate [1] just to make sure that there aren't any errors. For example, while writing this article I had accidently set some incorrect scaling factors in the Gerber output that became obvious in the Gerber viewer, and were easy to fix.

Bill of Materials

A Bill of Materials (BOM) lists all of the component information in a design so that it can be manufactured. DesignSpark includes the ability to generate BOMs as part of its Reports feature in the Output menu, as shown in **Figure 4**.

The built-in Bill Of Material report will generate a BOM with the following fields:

- Ref Name: The reference designator for the component
- Qty: The number of components for that line, always 1
- Component: The component name field
- Value: The component value field
- Package: The component PCB footprint type
- Manufacturer: The component manufacturer field
- MPN: The component manufacturer part number field

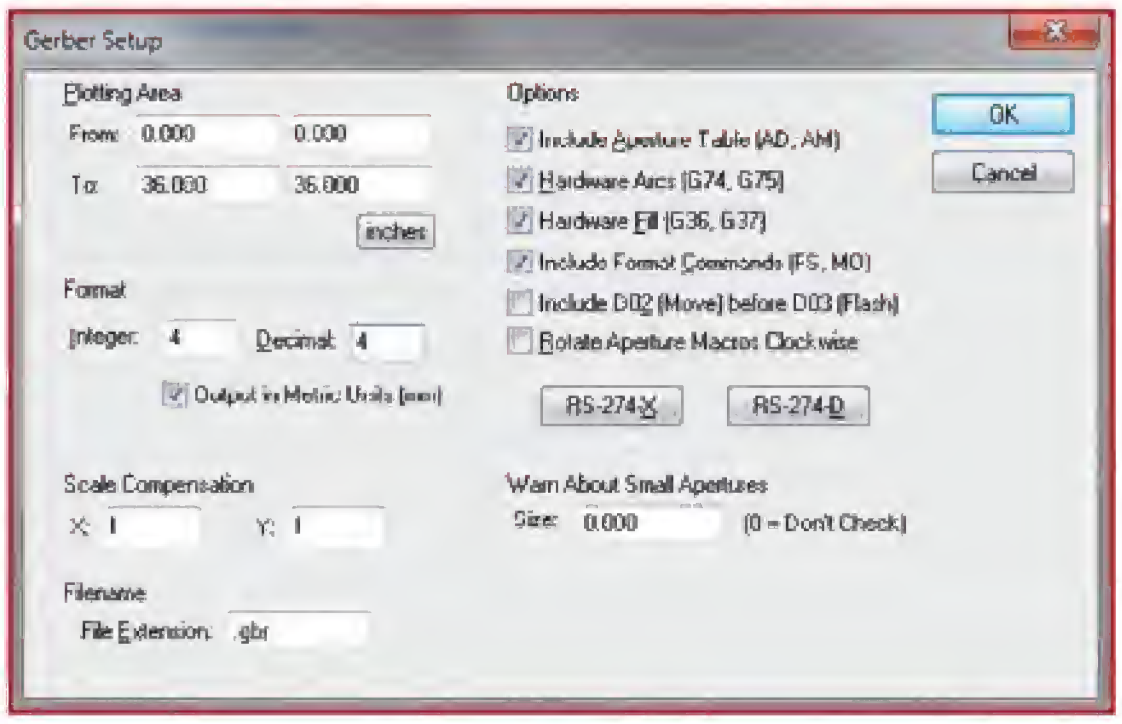


Figure 2.
Selecting metrics with 4-decimal accuracy.

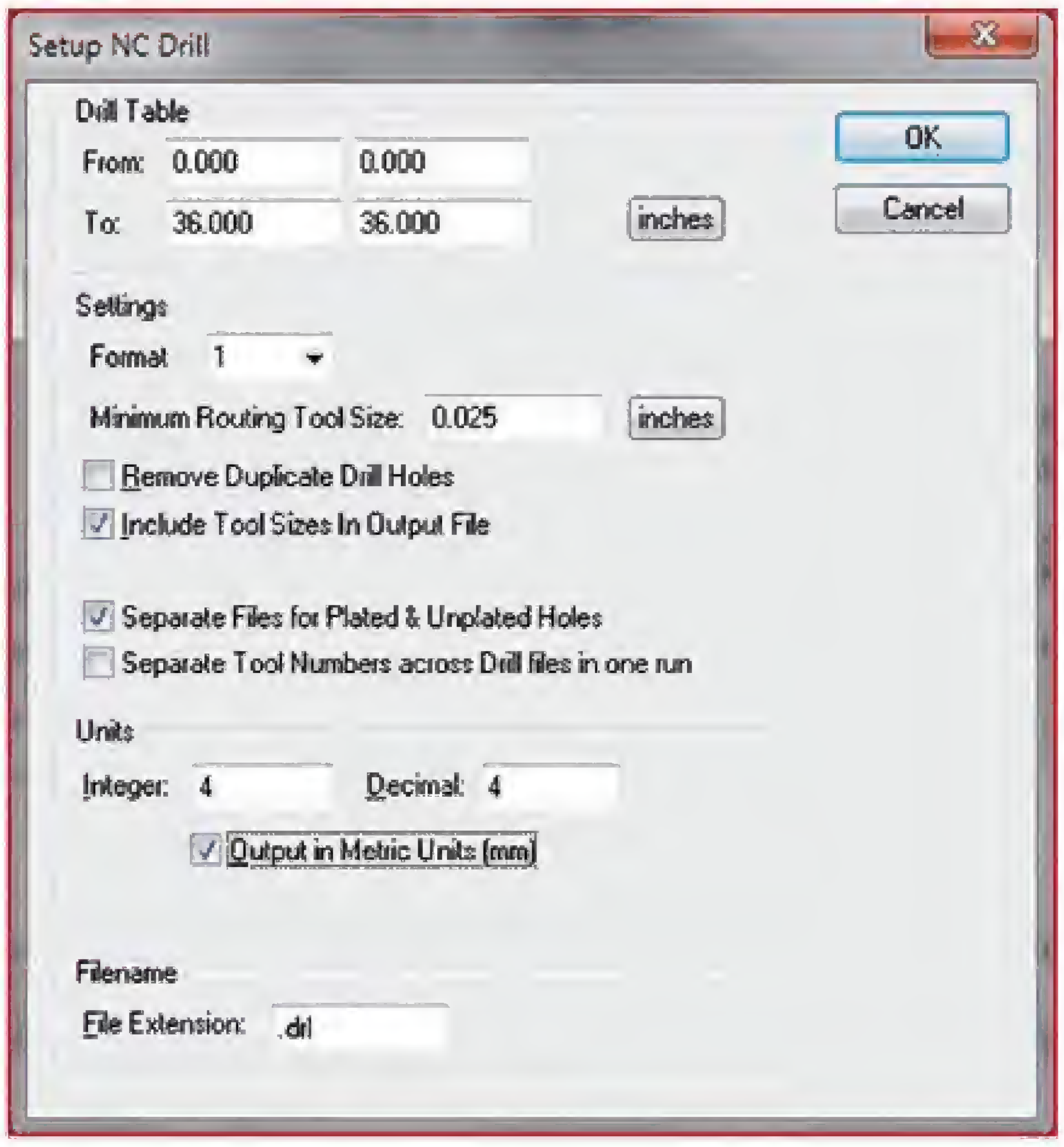


Figure 3.
Four-decimal precision is also defined for the NC drill data.

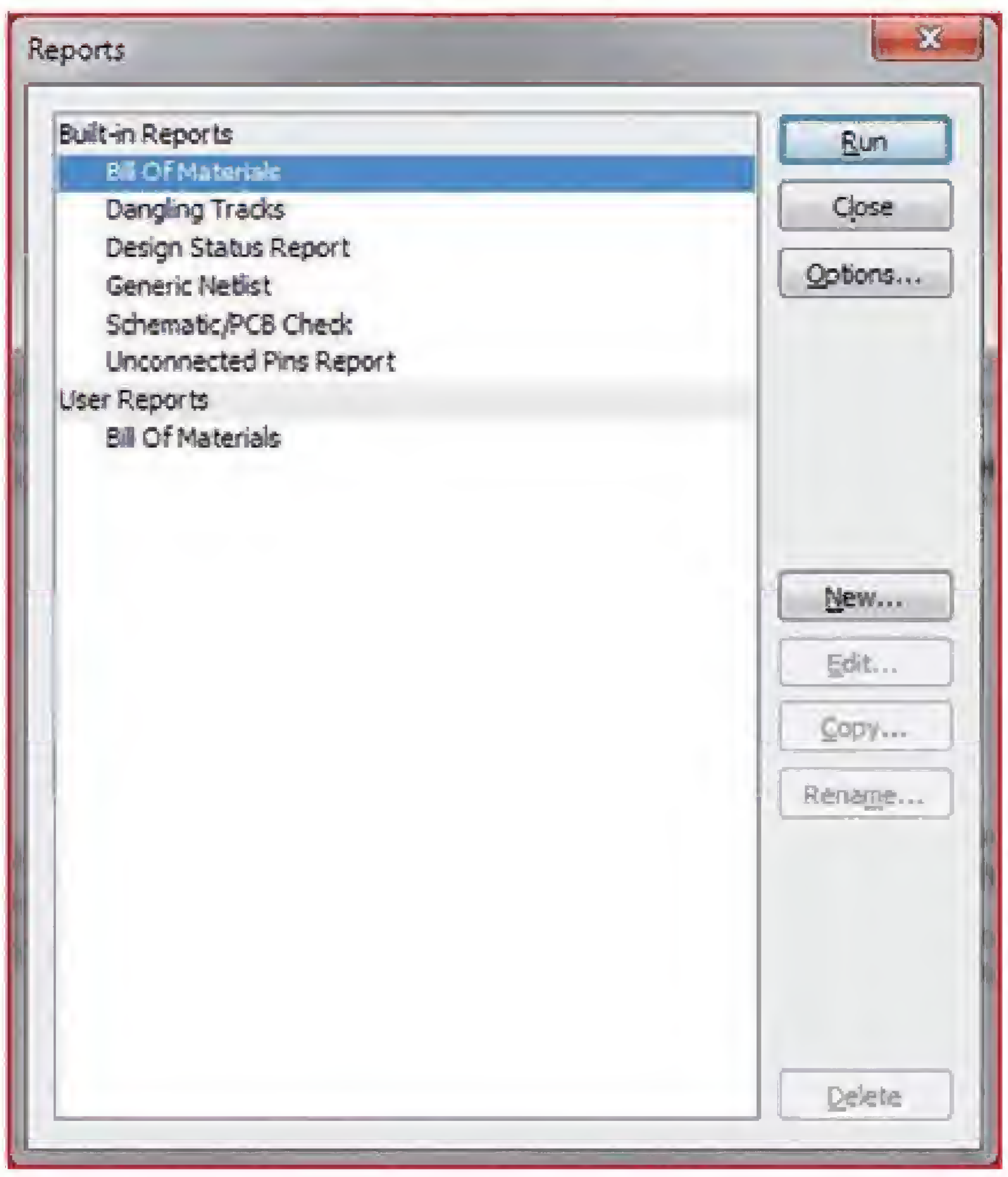


Figure 4.
Selecting the BOM as a Report.

- RS Part Number: The component RS part number field
- Description: The component description field

The default BOM is fine if you only have one part number per component, but I like to associate alternate part numbers to the component so that my BOMs have all that information available automatically. I also like to see the same part grouped together instead of a line for each component. For example, I find it better to know that there are two 1 K (kΩ) resistors in a design instead of having to count them manually in the BOM. Unfortunately DesignSpark cannot group components together on a BOM and display their reference designators in the same report, which means we will have to create two custom BOMs—one that is grouped for purchasing and another with reference designators for assembly.

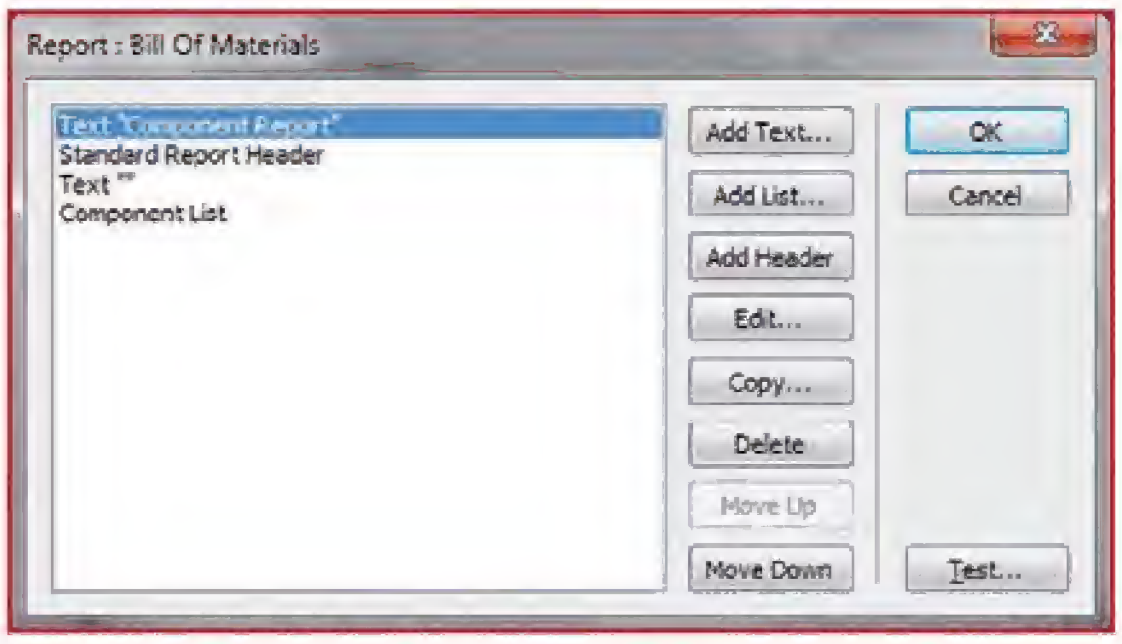


Figure 5.
Editing options for the BOM report generator.

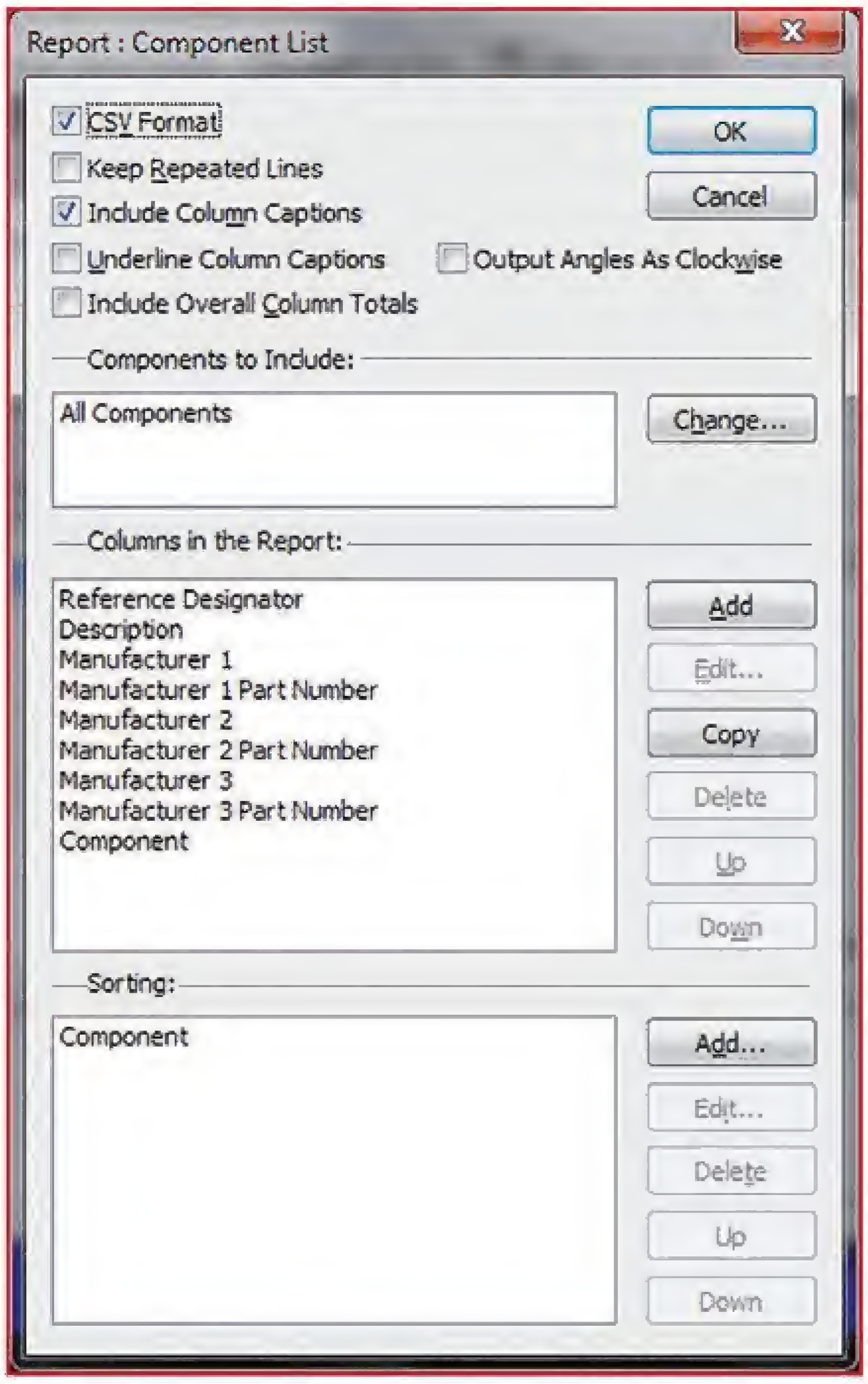


Figure 6.
One way of composing the BOM.

The first step is to create a new report by clicking on the New button. A dialog box will pop up where you can give the report a name, and then you will be able to edit the report content. It will look like in **Figure 5**.

What's happening here is that the first line in the report will be the text "Component Report", followed by DesignSpark's standard report header, a blank line and then a list of all the components. The part we have to edit is the Component List. For the purchasing BOM I edited the report columns to be Qty, Description, Manufacturer 1, Manufacturer 1 Par Number, Manufacturer 2, Manufacturer 2 Part Number, Manufacturer 3 and Manufacturer 3 Part Number. The assembly BOM is the same as the purchasing BOM, except that it includes the reference designators or Ref Names in DesignSpark. **Figure 6** shows the configuration I use for assembly BOMs.

To add custom report columns like Manufacturer 1 to your report you need to choose Value from the Field dropdown box. This will enable the Values box where you can select one or more component fields to be used for the column. The Caption field will be the column heading in the final report. It's also important to run the BOM reports from the PCB file and not from the schematic—to make sure the BOM is populated properly.

Conclusion

Today we generated Gerber files and a BOM from our DesignSpark design enabling it to be manufactured. Next time we'll look at some of the online quoting tools built into DesignSpark.

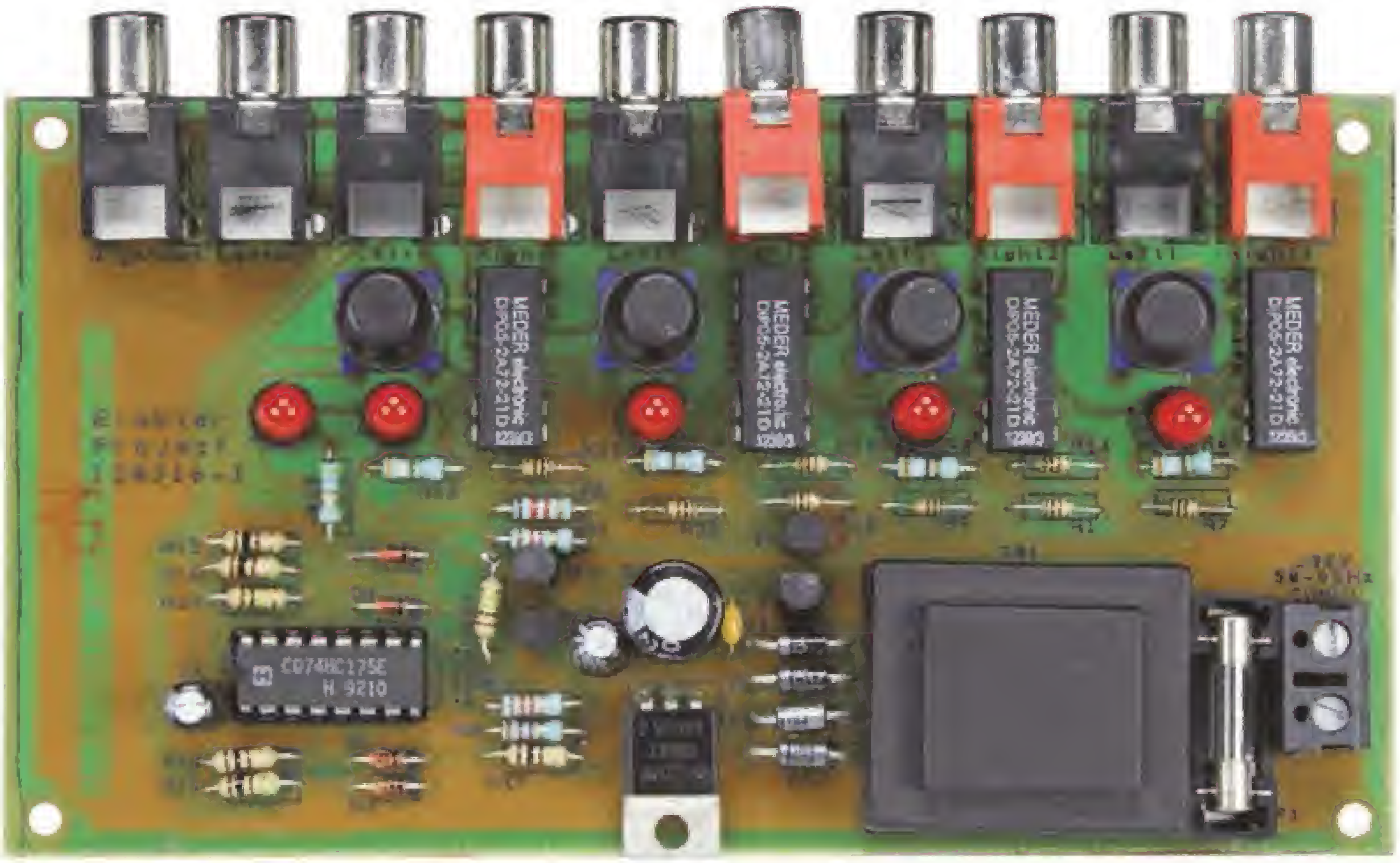
(130240)

Internet Reference

1. www.pentalogix.com/viewmate.php

Bidirectional Stereo Input Selector

1→4 or 4→1



Following a recent appeal* made in the magazine calling on our readers to send us their own circuit designs—if possible simple and preferably in fields that receive less coverage in Elektor, like audio, for example—we received this suggestion which meets the two main criteria: uncomplicated and intended for processing sound signals.

By **Olivier Croiset** (France)

In this era of the microcontroller heavily dominating the electronics scene, a circuit like this one may seem pretty simple, simplistic even—but it does have the undeniable merit of not frightening off someone just starting out in electronics! What's more, it lends itself to all sorts of modifications, particularly to extend the possibilities or adapt it to specific needs.

A good opportunity for beginners to stray from the beaten track, beginning with a circuit that's sure to work. The idea arose out of a practical need I had at home: repairing the faulty input source selector switch on my amplifier—a well-known and often thorny problem. These multi-way selectors are not usually standard types, and so are almost impossible to find.

With this little project I'm suggesting, I was also keen to reduce audio wiring (even screened), so as to keep it simple and above all minimize possible sources of interference.

Not for microcontroller fans!

The version of the diagram in **Figure 1** has been reworked and improved in the Elektor Labs and then redrawn in accordance with Elektor's house style for schematics. My original version was drawn using the DesignSpark PCB application. On the left, four buttons (one per channel) S1–S4 are associated with four diodes (D1–D4) to form a logic OR function. The output of this quasi logic operator is the common point of the four cathodes. This goes high when one of the buttons is pressed. This controls the clock (CLK) input

* this appeal is still open: your original contributions are welcome in the magazine. Put them on line on our community website www.elektor-labs.com; the authors of suggestions taken up by the editorial staff will be contacted with a view to setting up a publication contract.

a trouble-free audio add-on, suitable for beginners

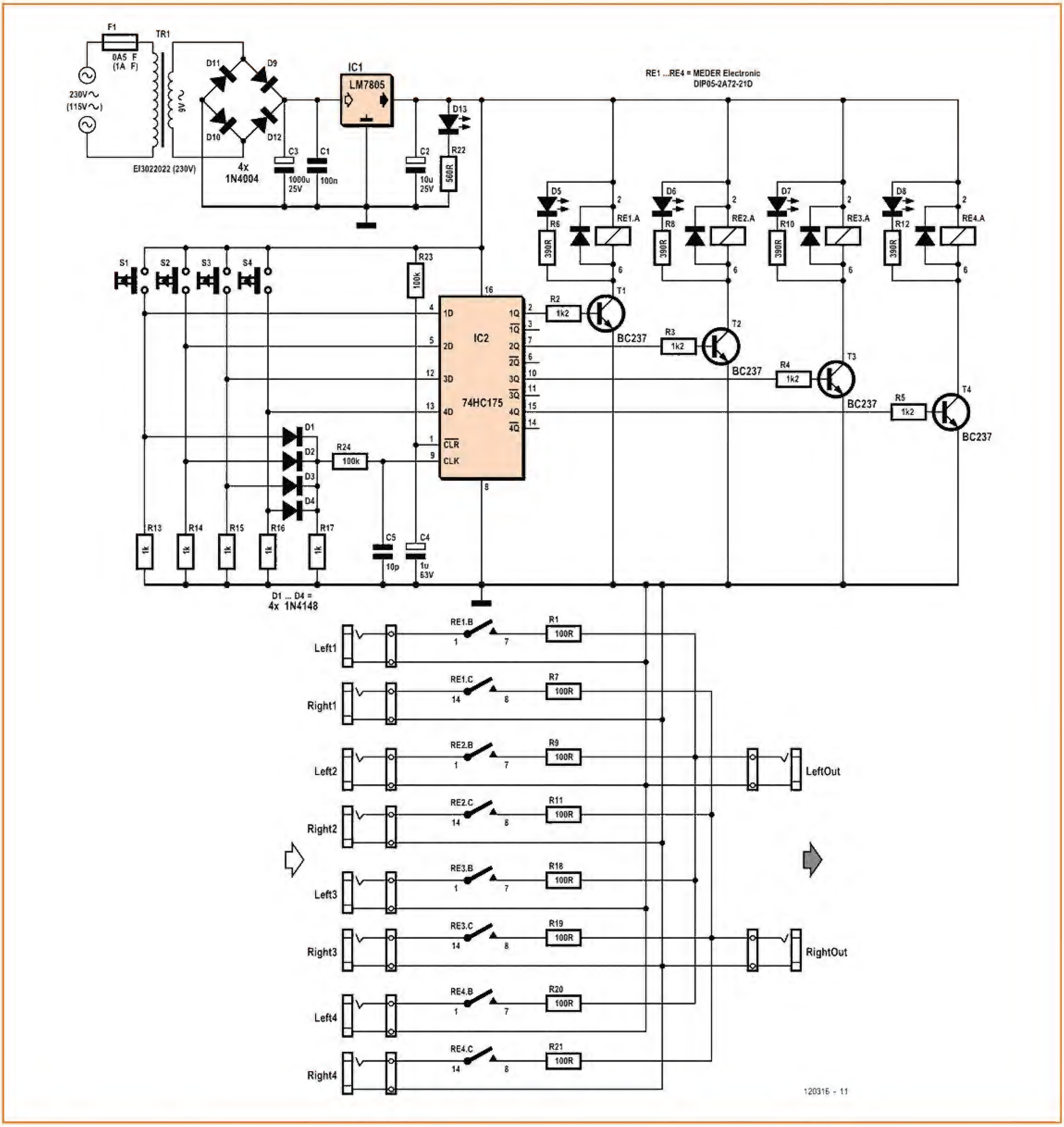


Figure 1. The audio source selector circuit is not directional: designed for four stereo inputs and one output, it can perfectly well be used the other way round.

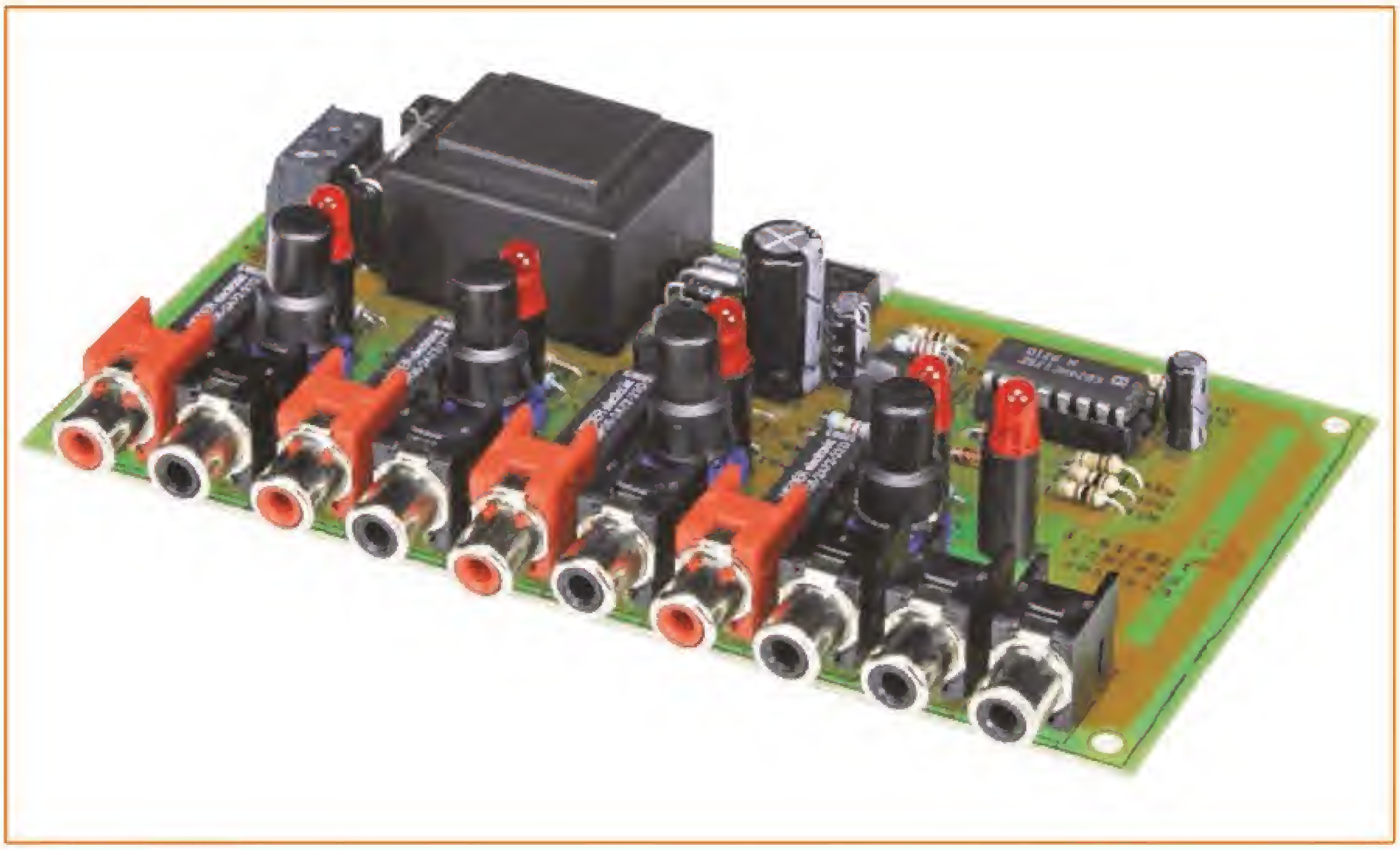


Figure 2.
The two outputs are distinguished by using two black sockets for the L and R channels, while the four R inputs use a red socket. Tip: note how the indicator LEDs are kept straight using the cylindrical spacer on which they are mounted.

common to all four D-type flip-flops in IC2, a 74HC175. Each time the CLK input goes high, the flip-flop Q outputs adopts the logic level present at this moment at their data input D (IC2 pins 4, 5, 12, and 13). When the button is released, the CLK input (pin 9) is immediately taken low. However, nothing happens, as the flip-flops only react to rising edges on the CLK input.

Pressing one of the channel selectors S1–S4 also takes the D input of the corresponding flip-flop high. The corresponding Q output (pins 10, 7, 10, and 15) likewise goes high. We are now in the right-hand part of the circuit diagram. The Q output of the triggered flip-flop in turn drives one of the transistors T1–T4, which turns on and activates the relay Re1–Re4 in its collector circuit. In this way, one of the audio signal channels is selected (lower part of the circuit diagram in Figure 1). The LED corresponding to the active channel (D5–D8) lights.

The snubber (back-emf) diode shown to the right of each relay is not a separate component, but is included within the device. Its function is to protect the transistor from the voltage spike that appears on the relay coil when it is de-energized. On the left, we have four left and right pairs of stereo signal sources (Left 1–4 and Right 1–4) and on the right a stereo output (LeftOut and RightOut).
Do note that even though we are referring to 'input' and 'output', this is no more than a convention, since the circuit can perfectly well be used in reverse, with one input (LeftOut and RightOut) feeding four sound equipment inputs (Left 1–4 and Right 1–4).

So what do you think happens if you press two buttons at once? Nothing, because the flip-flop outputs only adopt the logic level of their D input when there is a level change (rising edge) on the CLK input. So once a button is pressed and not released, this CLK input is kept high. There's no rising edge, so the flip-flops don't react.

At power-up, the $\overline{CLR}$ input to the four flip-flops is kept Low while C4 is charged through R23. In this way, the flip-flop Q outputs are forced Low, regardless of the level of their D inputs. At start-up, the four relays are always de-energized, only the indicator LED D13 is lit.

Among the possible variants of a circuit like this, we can note that depending on whether the transistor is driven from the flip-flop's Q or $\overline{Q}$ output, we can energize just one of the relays, as we are doing here, or all but one of the relays. The latter option is not catered for on the PCB as proposed here.

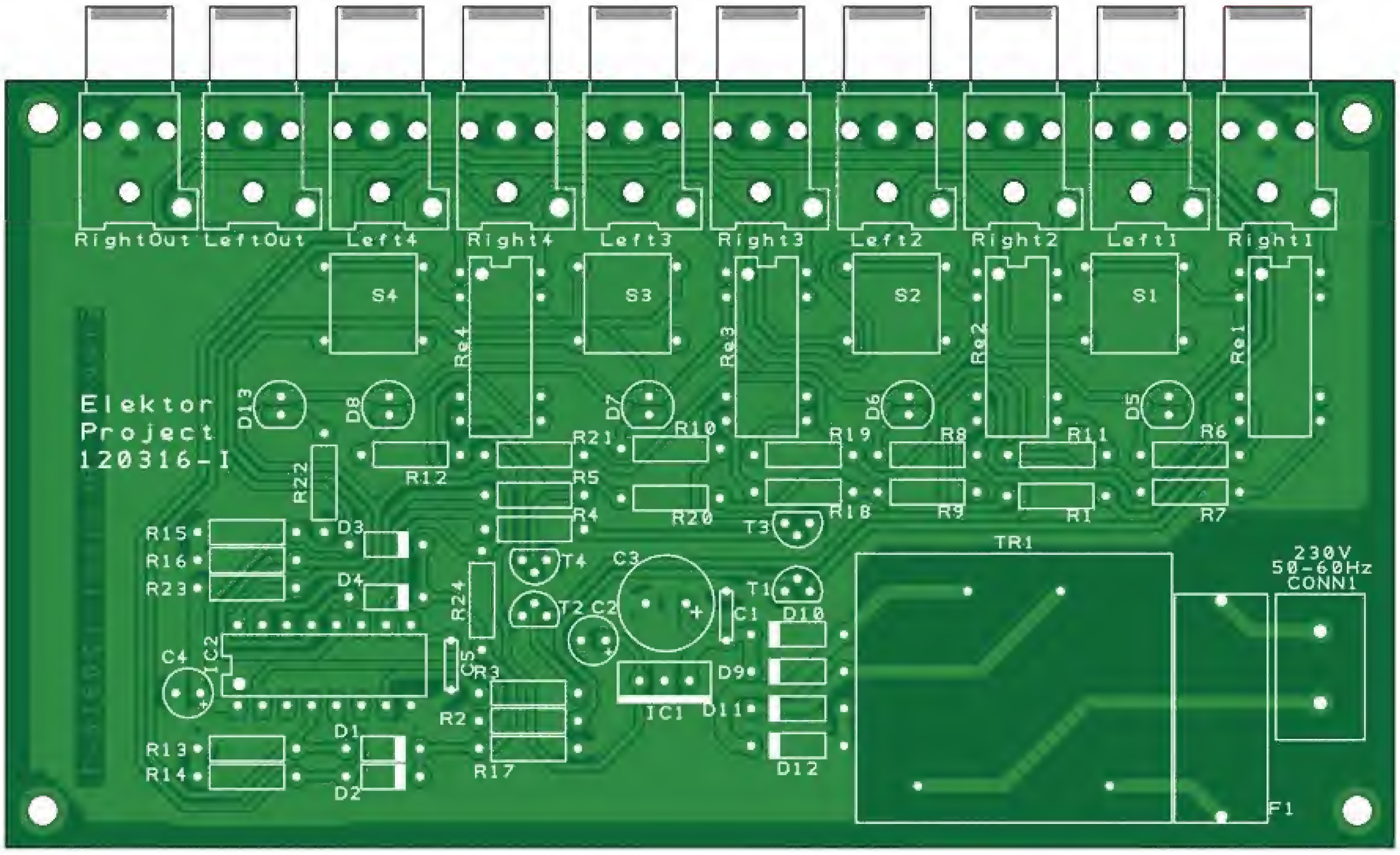


Figure 3.
Stereo channel selector PCB design. There's no need for a heatsink on voltage regulator IC1, but there would be space to fit one.

Selection, a sound trap

To make this stereo input selector easier to build, Elektor Labs have redesigned a single (superb) PCB using Design Spark, although I had originally planned on two separate boards; this makes it possible to have the relays and inputs/outputs some distance away from the buttons and LEDs, to which they would be connected using ribbon cable.

In this instance, we have a sort of remote control. It's debatable; both options have their drawbacks. In any event, even though my original configuration has not been adopted by Elektor Labs, I'm pleased that my little circuit is being published, and what's more as a "normal" article—I was only expecting it to get a quarter page in a Summer issue!

The circuit here is powered from the AC powerline, using a transformer for which both the safety and signal track spacings have been respected. However, if you have stricter requirements as far as electrical safety is concerned, and the presence of the transformer bothers you, it can be fitted away from the PCB, or even done away with altogether by powering the selector from batteries.

(120316)

COMPONENT LIST

Resistors (.25W, 1%):
R1,R7,R9,R11,R18-R21 = 100Ω
R2-R5 = 1.2kΩ
R6,R8,R10,R12 = 390Ω
R13-R17 = 1kΩ
R22 = 560Ω
R23,R24 = 100kΩ

Capacitors
C1 = 0.1 µF 100V, ceramic disc
C2 = 10µF 100V electrolytic
C3 = 1000µF 25V electrolytic
C4 = 1µF 63V electrolytic
C5 = 10pF 100V, ceramic disc

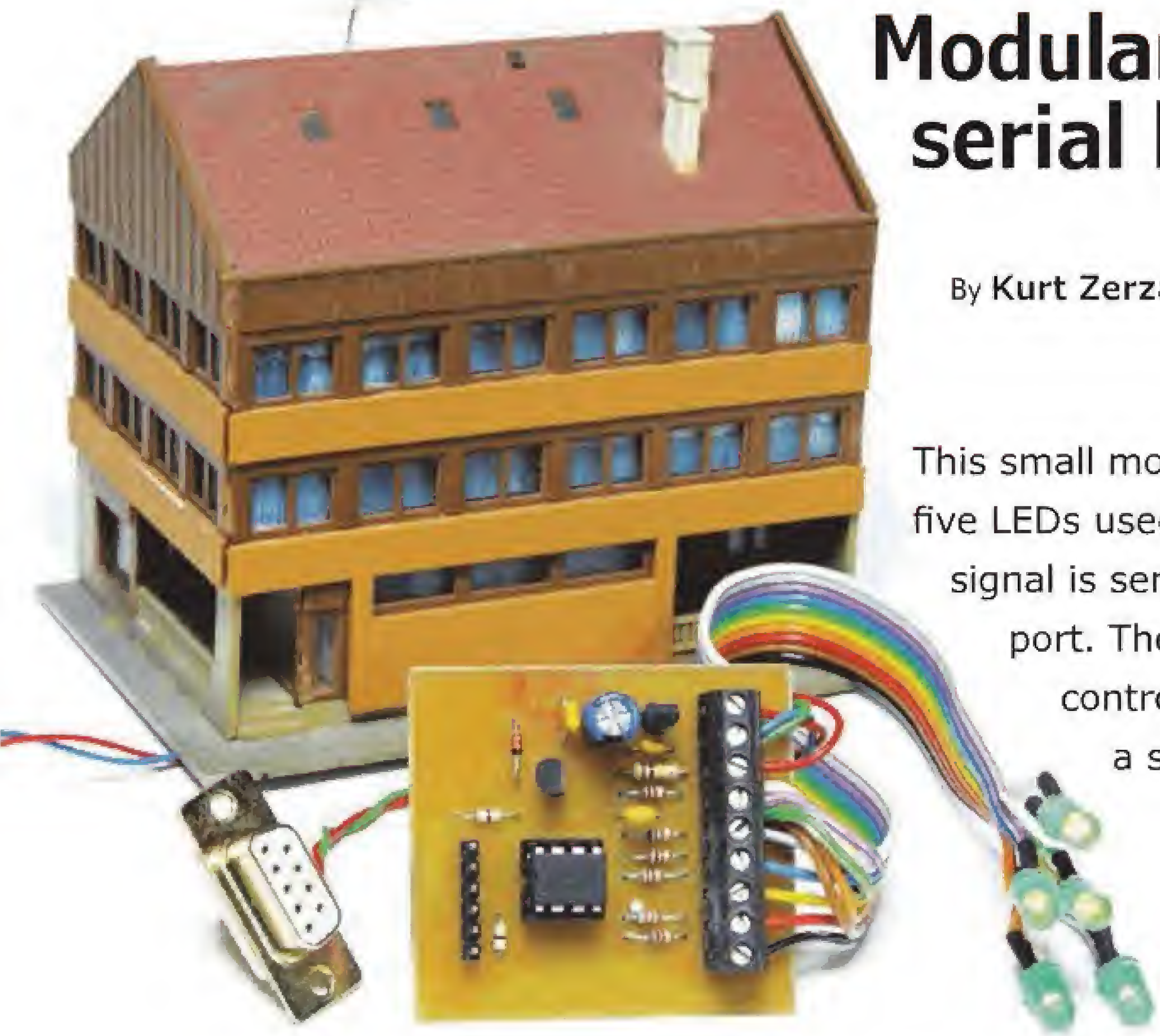
Semiconductors
D1-D4 = 1N4148
D5-D8,D13 = LED, 5mm, yellow
D9-D12 = 1N4004
IC1 = LM7805CT
IC2 = CD74HC175E

Miscellaneous
F1 = fuse, 500mA, fast
K1-K10 = RCA (line) audio socket, PCB mount
Re1-Re4 = relay, DIL, PCB mount, e.g. MEDER type DIP05-2A72-21D
S1-S4 = pushbutton w. make contact
TR1 = power transformer, secondary 9V @ 2VA

LED Lighting For Model Buildings

Modular, PC controlled on a serial link

By **Kurt Zerzawy** (Switzerland)



This small module has been designed to individually control five LEDs used to illuminate model buildings. The control signal is sent over a single wire from a PC's RS232 serial port. The design allows for up to 250 modules to be controlled by a single PC, that's almost enough for a small town!

Even if you only have a modest model train layout at home, the chances are that it will have a station and some other buildings. Rather than switching all the house lights together, it gives a much more realistic appearance if the lights can be individually controlled. In reality the light pattern changes as occupants move from room to room. The serial control module achieves this while avoiding the dreaded cabling rat's nest. Its modular design allows you to start small and easily add more modules as the layout gets more ambitious.

Cutting costs

The circuit does not need to perform any demanding tasks—for this application the main design criteria are the module's size and cost. In fact the circuit shown in **Figure 1** consists of little more than a small 8-bit microcontroller and a voltage regulator. The microcontroller is a small 8-pin PIC12F675P from Microchip, in a DIP outline which can be found for not much more than a dollar. It contains a 1 KB program memory, 128 bytes of EEPROM and (essential for serial communication) a timer/comparator. The EEPROM is used to store the module's address. All of the pins can be defined as I/Os (except the supply pins naturally), the internal oscillator is used to supply the system clock. Download the data sheet [1] to find out more about the chip.

The anodes of the five LEDs for each module are wired to the PCB-mounted terminal blocks K1 and K2 which connect back to I/O pins GP0 to GP2, GP4 and GP5 on the controller chip. The cathodes of all the LEDs are connected together to pin 3 of connector K1. The 1-kΩ series resistors limit the forward current through the LEDs to safe levels. The current will be dependent on the type and color of LED used here. The forward voltage drop will typically lie somewhere between 1.9 and 2.5 V resulting in a current between 2 and 2.5 mA. It is important to use low-current LEDs here in order to achieve full illumination from just 2 mA supply current.

The LEDs are switched on and off by information contained in the signal received over the serial interface (via pin 1 of K3). This works essentially as an RS232 interface using just the receive signal (RXD) connected in parallel to all modules. The level shifter network D1 and T1 limits the ±12 V signal from the PC to +5 V at each module's input.

Voltage regulation is performed by a 78L05 regulator which is good for up to 100 mA. The 'usual suspects' can be seen lurking around this IC: capacitor C4 and the two 100-nF capacitors C1 and C2. The circuit can be powered from an unstabilized power adapter with an output between 8 and 12 V DC. The simplest option is to use a power adapter connected to K3. Each module requires just 15 mA, so it is possible to power a lot of modules from quite a modest adapter. When a really large number of modules is used it would be sensible to power them all from an external power adapter providing a stabilized 5 V at the necessary current. The on-board voltage regulator can then be left out of the circuit, replaced by a wire link between the input and output pad.

The pinheader strip JP1 is for the connection to a microcontroller programmer. It is not necessary if the PIC has already been programmed (or has been bought pre-programmed). Programming each module's address occurs over the serial interface and does not require this interface. The pin assignment of connector JP1 is given in **Table 1** which conforms to the Pickit2 [2] ISP programmer connector. During the programming procedure it is important to ensure that the chip is powered either from its on-board supply or from the Pickit2 programmer, not both at the same time!

Table 1. The Programming Adapter JP1

JP1	PIC12F675	Signal
Pin 1	Pin 4: GP3/MCLR	Vpp
Pin 2	Pin 1: VDD	+ 5 V
Pin 3	Pin 8: VSS	GND
Pin 4	Pin 7: GP0/AN0/**icspdat**	Data I/O
Pin 5	Pin 6: GP1/AN1/**icspclock**	CLK
Pin 6	–	–

Communicating with the module

The serial interface is used to send switching information to the LEDs and also to assign an address to the module.
During power up the module uses its LEDs to indicate if it has already been assigned an address: when all five LEDs briefly come on together then no address has been assigned, when they switch on in a sequence one after the other then the address has already been assigned. Serial communication occurs over the RS232 interface at 9600 Baud, no parity, 1 stop bit. These communication parameters can set up in the terminal emulator program on the PC. In HyperTerminal (Windows) select File/Properties/Configure to make the changes.

Module addressing

To assign an address to a module and store it in the PIC's internal EEPROM enter the following sequence:

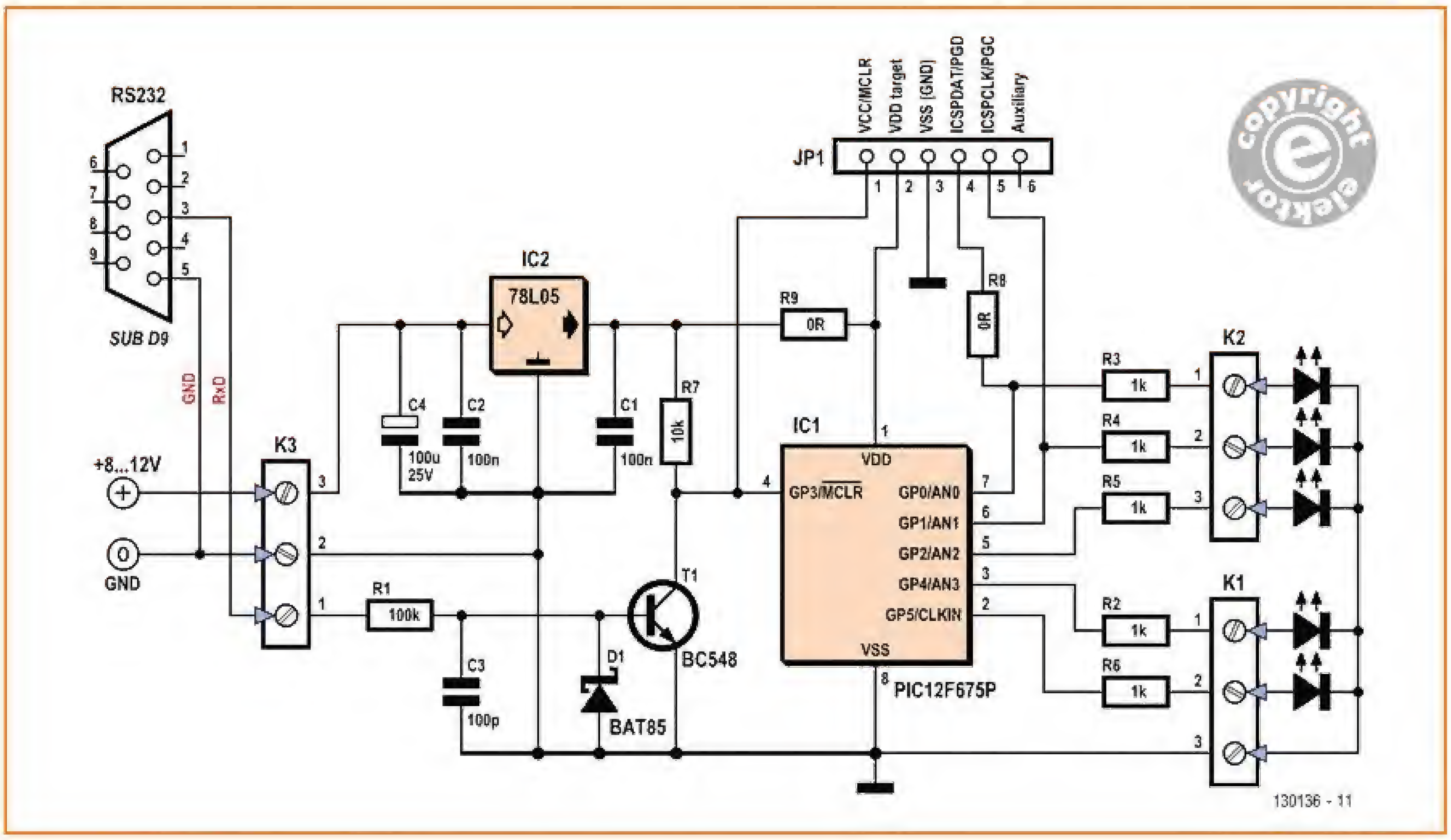


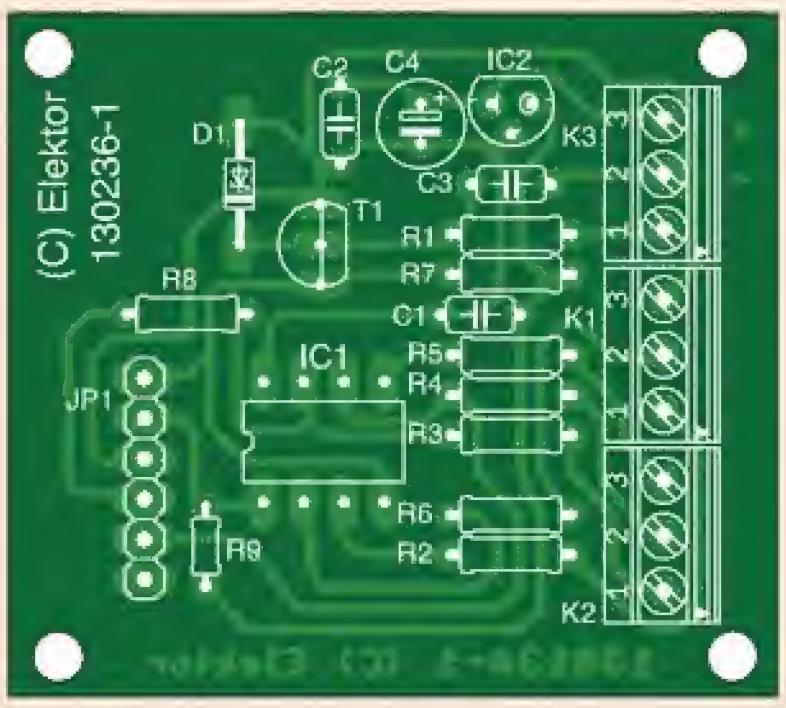
Figure 1.
The LED control module schematic.

COMPONENT LIST

Resistors
R1 = 100kΩ
R2–R6 = 1kΩ
R7 = 10kΩ
R8,R9 = 0Ω

Capacitors
C1,C2 = 100nF
C3 = 100pF
C4 = 100µF 25V (e.g. Rubycon 25YXF100MEFC6.3X11)

Semiconductors
D1 = BAT85
T1 = BC548
IC1 = PIC12F675-I/P (Microchip), programmed, Elektor Store # 130136-41
IC2 = 78L05



Miscellaneous
JP1 = 6-pin pinheader
K1,K2,K3 = 3-pin PCB screw terminal block
PCB # 130136-1 [3]

Figure 2.
Thru-hole components but a tidy layout nevertheless.

'H': House
'F': Hex address High-Byte (F for unprogrammed chip).
'F': Hex address Low-Byte (F for unprogrammed chip).
'P': Command P for programming.
'a'
'a'
'5': Safety code required to prevent accidental programming.
'5': Safety code required to prevent accidental programming.
'x': High byte (Hex) of the address to be assigned to the module, x = 0 to F.
'y': Low byte (Hex) of the address to be assigned to the module, y = 0 to F.
'CR': Carriage Return to terminate the sequence.

Entering the sequence 'HFFPaa55' at the terminal will cause the chip to switch on all the LEDs. Enter next the desired address (xy) and finally the 'enter' key. The LEDs will now go out and after an off/on power cycle the address will be stored. The same sequence can be used at any time to assign a different address to the module.

Example: To allocate the address 23h to a module that has not yet had its address assigned, enter the sequence 'HFFPaa5523' followed by 'enter'.

Switching LEDs

Controlling the module
'H': House
'x': Hex address high byte of the selected house x = 0 to F
'y': Hex address low byte of the selected house y = 0 to F
'S': Command S for set
'a': High byte of the selected LEDs (0 or 1)
'b': Low byte of the selected LEDs (0 to F)
'CR': Carriage Return (enter) to end the sequence.

The input sequence is therefore HxySab followed by 'enter', where xy is the addressed module, 'S' represents set and 'ab' is the hex value of the LEDs that will be turned on. This value can be in the range 00 to 1Fh, allowing every combination of the five LEDs to be switched.

Example: To turn on LEDs 3 and 5 of module 12, enter the line H12S14 followed by 'enter' at the terminal.

Build it modular

Figure 2 shows the PCB for a single module which can be ordered from Elektor. The use of thru-hole components simplifies construction greatly. The two 0 Ω resistors R8 and R9 are necessary to bridge some PCB tracks and can be replaced by wire links. Use a socket to mount the controller. All the modules are connected in parallel. Resistor R1 in each module ensures that only a very small amount of current is taken from the RXD signal, allowing a large number of modules to be connected without any problem. The RXD signal is generated from pin 3 of the 9-way sub D connecter on a PC. The earth connection is on pin 5 of the same connector and is linked to the earth wire in the power cord.

The pre-programmed microcontroller reference number 130136-41 can be purchased from the Elektor Store. Alternatively you can program the chip yourself if you posses the programming tools. This option is more attractive if you have a large layout requiring many modules. The software is free to download from [3] and includes all the CAD files for the project, viewable using the free CAD program DesignSpark.

(130136)

Internet Links

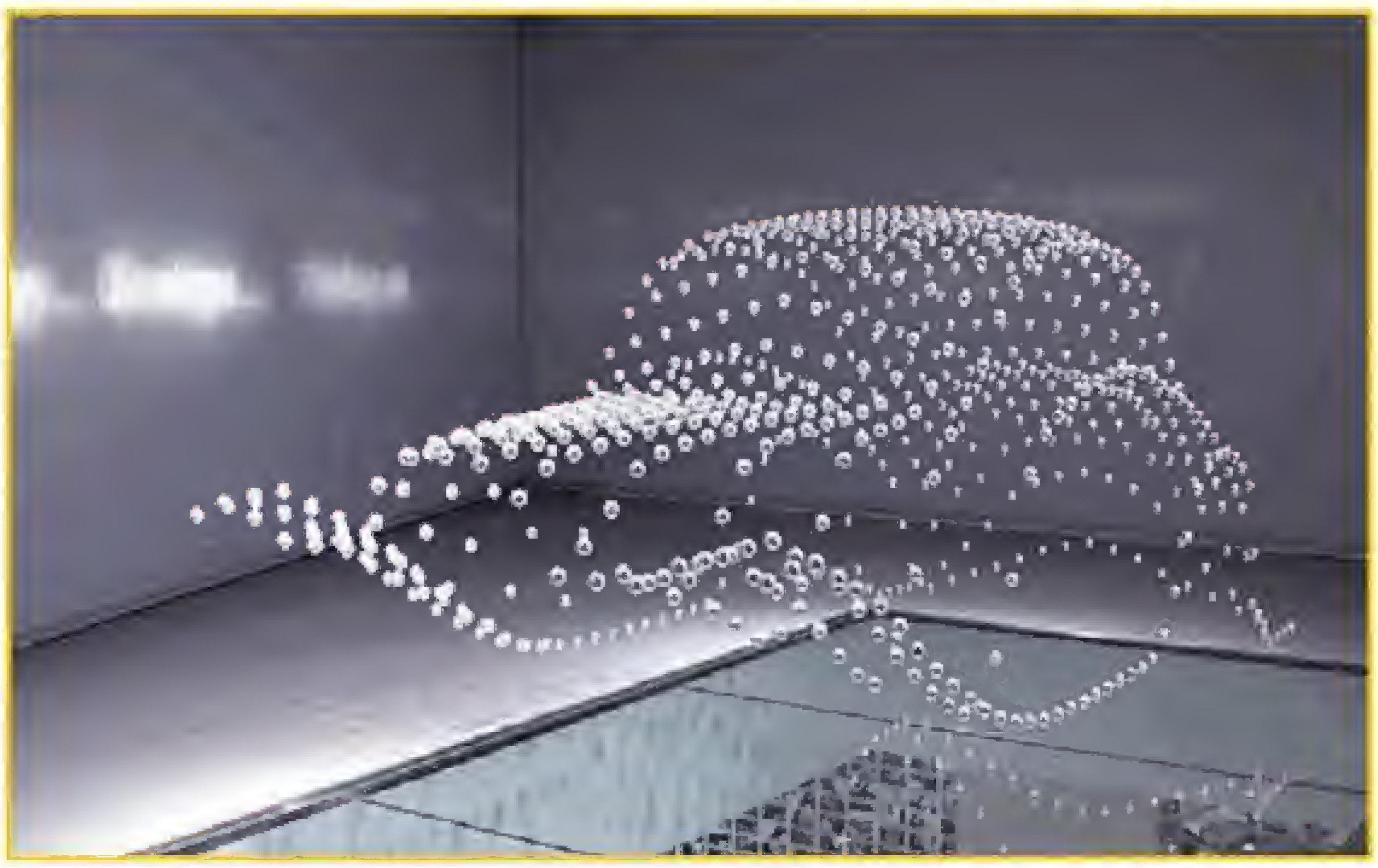
[1] ww1.microchip.com/downloads/en/DeviceDoc/41190G.pdf
[2] www.microchip.com/pickit2
[3] www.elektor.com/130136

.Labs Tips & Tricks

By Clemens Valens
(Elektor.Labs)

Some people have sound and/or stimulating ideas but do not know how to translate them into working electronic circuits, while others lack the skills and/or the time to complete a project. Below is selection of projects posted on Elektor.Labs and looking for a helping hand to reach the finish line. Can you cheer and/or assist?

Electronic Sculpture



Original Poster (OP) *snap* wants to create an electronic sculpture capable of producing figures in 3D by individually moving up and down some fifty balls attached to strings. There are some nice videos on YouTube showing such contraptions, just search on "BMW kinetic sculpture" or words of similar meaning. Winding and unwinding the strings can be done with small motors, but how do you control fifty of them in a synchronized way without spending too much money or time? The weight of it all is an important factor too. The OP is looking for help; do you have experience with a similar project? Are you confident with electronics, motors and mechanics? If so, why don't you surf over to www.elektor-labs.com/node/3450 and post a contribution.

Photo: BMW Welt

Five Cool Projects

Outdoor Solar Wireless Wi-Fi/3G Webcam
www.elektor-labs.com/node/3538

All-You-Can-Eat Bluetooth
www.elektor-labs.com/node/3032

PWM Controller For flashlight
www.elektor-labs.com/node/3537

RS-485 Sniffer
www.elektor-labs.com/node/3522

Mouse-Friendly Mousetrap
www.elektor-labs.com/node/3433

GMR-based Current Probe

Measuring current with a multimeter is easy; doing the same with an oscilloscope is a bit more involved. You can of course buy special current probes, but they tend to be expensive, which is why OP *RolandSautner* decided to design his own. Unexceptional current probes use Hall-effect sensors and AC transformers to measure DC and AC currents, but the OP wanted to use another method: giant magnetoresistance, or GMR. Why? Because he had an unused KMZ51 magnetic field sensor lying around and felt that it was a shame not to use it. The OP got pretty far, but finally encountered some problems mainly related to mechanics. Can you help the OP to transform his design into a practical probe? If you can, please go over to www.elektor-labs.com/node/3423 and post your ideas and suggestions.

Photo: Agilent

Frequency Inverter

This is about a project that has my personal interest, but I've never gotten round to building one. As always, I am not alone as I saw OP *PappaBaer*'s post. A frequency inverter or, to be more correct, a variable-frequency drive (VFD), is a device that can control the rotation speed of an AC motor—usually three-phase. Elektor has published such a device in 1994/1995. The project was very successful right up to being recalled again in *Retronics* in 2006, but it never saw a successor. The components used in the original design are now hard to find or even obsolete, and a redesign would be welcome. This time with open-source software, of course. Have you ever been involved in such a project? If so, please join us at www.elektor-labs.com/node/3484.

Photo: Wikipedia

www.elektor-labs.com

Trailer Reversing Lamp

One of the great things about Elektor.Labs is the wild diversity of the projects posted. There are lots of things to discover thanks to the different interests of the authors. Some time ago *PLEG54* posted a project to develop an adapter for old vehicles allowing them to pull modern trailers sporting a reversing lamp or back-up light. ECE regulation R48 imposes this. ECE-R48? I did some research and discovered The World Forum for Harmonization of Vehicle Regulations. This is a working party of the Inland Transport Division of the United Nations Economic Commission for Europe (UNECE), assigned with creating a uniform system of regulations for vehicle design to facilitate international trade. Regulation R48 concerns the installation of lighting and light-signaling devices. This means that the UN decides which lights and reflectors you need to have on your car and trailer, and where. Anyway, this is all very instructive, but the really interesting part is that the OP is stuck and can use some help. You can find him or her at www.elektor-labs.com/node/3360.

Photo: Wikipedia

Note: OP stands for Original Poster, the person who started an online project or discussion. OPs who want to have a chance of appearing in Elektor Magazine must (regularly) check the email address they use to access Elektor.Labs. This is our only means of contact.

Standards for Coding

Power your circuit with better software

By **Clemens Valens**
(Elektor.Labs)

Often a lot of time and energy is spent on designing an elegant, well-thought-out and robust circuit. Today, the brains of many of those circuits is a microcontroller that needs software to function. Is it unreasonable then to expect a well-designed, properly written program to make such a quality circuit work? Apparently it is. Let's talk software quality.

It is a well-known fact: bugs kill people and software bugs are no exception. Every year people die because of failing software. Some fatal airplane, helicopter and car crashes can be attributed to software problems; malfunctioning medical device firmware make victims on a regular basis; buildings go up in smoke due to bugs, and some people get hacked to death. Bad software can even sink boats. Believe me, I have seen it happen.

A bug-free piece of software doing a meaningful job does not exist. According to Wikipedia, NASA's Software Assurance Technology Centre has managed to produce software with less than 0.1 bugs per 1000 lines of code, which is considered to be extremely good. Commercial organizations do not have the time and money to achieve such a level of quality. Apparently Microsoft tries to be at 0.5 bugs per 1000 lines when it releases a new product, so realizing that Windows XP was compiled from 45 million lines of code, you know that it contained more than 22,500 bugs when it was released. It is estimated that programs written in industry for internal use have error rates from 5 to 50 bugs per 1000 lines.

Software defects—a more official term for 'bugs'—can have many origins, from badly understood, complex problems to downright sloppy programming. Contrary to popular belief, making good software is not easy as it demands extreme precision and care. For every line of code that must be written, at least three things are required:

1. a good understanding of the problem and its solution that the line of code is supposed to implement;
2. the capability of expressing this solution correctly in the programming language at hand;
3. not making any typing errors while entering the code.

The first point is often the hardest part, especially for large projects. This is why skilful system architects and good specifications are essential to the success of a project.
Item 2 is also a difficult one as it involves the choice of the programming language—not every language is suitable in every situation—and expressing logic reasoning in a non-natural language without making mistakes.
The third point seems moot, but is actually pretty hard to get right. It is very easy to miss a brace or a bracket, swap two symbols or mistype a character. It is only at this point that the programming tools start helping the developer and only by pointing out syntax errors and maybe compatibility issues and other conflicts between data objects.

To reduce the risk of creating future defects due to our line of code, we can add to the list:

4. make sure the code is written in a clear and understandable way;
5. add comments (and keep them up-to-date) to explain the reasoning behind the line of code;
6. adhere to a coding standard.

Item 4 refers to the use of comprehensible names for functions and variables. Although it may require more typing, a variable name like

Windows XP was compiled from 45 million lines of code – on release it contained more than 22,500 bugs

"acceleration" is much more explicit than "acc" or just "a".

Item 5 is ignored by many programmers. Not because they are not aware of the comment feature of the programming language they use, but because they are too lazy to use it.

The last point is a very important one. Many amateur programmers are not remotely aware of coding standards, while some professional programmers are, but don't care.

But what is a coding standard?

Most software producing companies have house rules or conventions on how their developers should write code. On the internet you can find coding standards for many open source projects like GNU [1] or Linux. However, most of the time these rules are limited to the formatting of the source code, and their goal is to create uniform source code which can be maintained more easily. Since they concern mostly the appearance of the code, it is better to speak of a coding or programming style.

A real coding standard is not a set of rules to make your program look pretty; it is a set of rules to reduce programming errors. It may also be used to achieve compliance with a regulatory standard. Because today C and C++ are the prevailing programming languages, most coding standards address C/C++. The advantage of conforming to a standard is that you can use static analysis tools to check the semantics of your code instead of only the syntax, which helps you to identify possible issues a compiler cannot find.

So, what kind of rules can you expect from a coding standard? Here are a few examples:

Ban explicit language features that can hide coding errors

In C and C++, it is legal to do use the assignment operator '=' in Boolean expressions. For example, the following expression is legal:

```
if (sample=get_sample()) sample += 2;
```

This line of code will add 2 to `sample` if `sample` is not equal to 0. Why? Well, first the function `get_sample()` is called and the value it returns is assigned to the variable `sample`. Now the Boolean `if` statement will check if the condition between its parentheses is true or false. In C/C++, false equals 0 and true equals not false. Therefore, if `sample` is non-zero, the condition is considered to be true and the addition will be executed.

But maybe the programmer intended this:

```
if (sample==get_sample()) sample += 2;
```

The difference, an additional '=' character, is subtle. In C/C++, the sequence "==" means "is equal to". So, the value of 2 will be added to `sample` only if the value of `sample` is equal to the return value of the function `get_sample()`. This is clearly not the same behavior as before. Is this a programming mistake or was it intended? Impossible to know. Some compilers can flag this issue, but only if the warning is activated. This is why a coding standard bans this kind of language features.

Year 2000, Y2K or Millennium bug (1999)

Due to the practice of abbreviating a four-digit year to two digits, many programs risked to calculate wrong dates after 1999. Huge efforts were made to prevent the bug from biting. Worldwide costs are estimated at a whopping 425 billion dollars, but in the end nothing serious happened. Something similar may happen on the 19th of January 2038 when the UNIX seconds counter will overflow.

Only use initialized pointer expressions

This is a classic pitfall and the cause of innumerous bugs. When a pointer to a data object is not initialized—i.e. a 'wild' pointer—it can point to anything. Using such a pointer will result in undefined behavior of the program. Again, some compilers can flag this issue, but only if the warning is activated. Forbidding the use of uninitialized pointers is the only solution. Because pointers

Ariane 5 Flight 501 (1996)

A complete loss of guidance and attitude information 30 seconds after lift-off due to specification and design errors in the software of the inertial reference system caused the satellite launcher to disintegrate. The origin of the crash was a 64-bit floating point value that did not fit in a 16-bit signed integer, resulting in an overflow. The unexpectedly high value was calculated by an algorithm designed for the Ariane 4. Financial loss was estimated at some 400 million dollars.

are dangerous, some other programming languages restrict their use.

Pointers are closely related to buffer overflows, number 3 in the 2011 CWE/SANS Top 25 of most dangerous software errors (that deals mostly with code security issues).[2]

```
int* p_some_pointer;
p_some_pointer = address_of_data_object;
p_some_pointer[34] = 3;
```

Although in the above code fragment the pointer is initialized before being used, one question remains: is the index of 34 valid? If not, it is a buffer overflow error.

Strong data typing

In C/C++, variables of one type can be assigned to variables of another type as long as the new data type has the same or better precision. For example, assigning an integer variable to a floating-point variable will not produce a warning. Going the other way may generate a warning. This means that for instance a character represented by an 8-bit value can be added to a 32-bit floating-point value; the compiler takes care of the conversion. To prevent adding pears and apples, a coding standard will forbid the programmer to mix data types unless an explicit type cast—changing a data type into another—is provided, like so:

```
float a = 3.14;
int b = (int)a;
```

Eliminate unused or unreachable code

Normally all code lines of a program have a function. However, it can happen that, due to a programming error, one or more parts of the program become unreachable because the execution path to those parts is cut off. Take this Arduino sketch:

```
void setup(void)
{
  int a = -1;
  unsigned int b = 1;
  if (a<b) a += 2;
  Serial.begin(115200);
  Serial.println(a);
}

void loop(void)
{
}
```

What value will the serial monitor print for the variable `a`? −1! Why? Because, in C/C++, if in a comparison one of the values is of type `unsigned`, the other value will be silently 'promoted' to `unsigned` too. But a 16-bit integer—as in Arduino—holding a value of −1 (i.e. 0xffff in two's complement notation, the way negative values are represented by most processors) is bitwise identical to an unsigned integer holding 65,535

Intel Pentium Floating-point Divide (1993)

A design error in Intel's brand-new Pentium microprocessor causes small errors when floating-point numbers that occur within a specific range are divided. Although the bug affects few users, it becomes a public relations nightmare and ultimately costs Intel some 650 million dollars.

(0xffff). Since 65,535 is larger than 1, the condition will always evaluate to false, and the addition is never executed—it is unreachable. Requiring that the programmer removes unreachable code will either force him/her to rethink the algorithm or to simplify the program. Remember, less code means fewer bugs.
This example can be made compliant to the coding standard either by removing the if-statement, or by applying the strong data typing rule to variable b, i.e. adding a typecast to temporarily make it a signed integer:

```
if (a<(int)b) a += 2;
```

Place limits on complexity

Long functions tend to be complex, and that makes them difficult to comprehend and test. For this reason a coding standard may forbid that a function or method contain more than, say, 200 lines of code.
Coding standard verification tools can use other metrics to measure complexity. An example is cyclomatic or conditional complexity that measures the number of independent paths through a function. The more paths there are, the higher the function's complexity. This kind of complexity is difficult to estimate for a human being—let a tool do it for you.

Style

Although I said earlier that the purpose of a coding standard is not to make your code look pretty, code may include styling rules to make source code easier to read. Easy-to-read source code makes identifying potential defects easier too. Examples of such rules are:

- Source lines will be kept to a length of 120 characters or less;
- Each expression-statement will be on a separate line;
- Tabs should be avoided;
- All indentations will be at least two spaces and appear consistent within the same source file;
- The statements forming the body of an `if`, `else if`, `else`, `while`, `do-while` or `for` statement shall always be enclosed in braces, even if the braces form an empty block;
- Braces ("{}") which enclose a block will be placed in the same column, on separate lines directly before and after the block.

Many programmers offend against these rules; some programming editors do too by automatically inserting tabs. BTW, why avoid tabs? Because they mess up source code formatting when two or more persons do not use the same tab distance.

Habits

C/C++ compilers come with a large collection of so-called standard libraries. Many programmers rely on the availability of these libraries, and use them out of habit. Unfortunately, several functions included in these libraries may exhibit

Popular coding standards

Not counting in-house standards, today the most widespread coding standards are (in order of popularity):

MISRA C (and C++) — created by the Motor Industry Software Reliability Association to provide assistance to the automotive industry in the application and creation within vehicle systems of safe and reliable software. A small fee must be paid to obtain the standard.
www.misra.org.uk/

CERT C++ (and C) — this initiative from the Software Engineering Institute of the Carnegie Mellon University strives to eliminate insecure coding practices possibly leading to vulnerabilities that may be exploited by malicious entities.
www.cert.org/ (note the use of the http secure protocol https. CERT is secure all the way.)

HICPP — High Integrity C++, made freely available by PRQA, provides guiding principles for maintenance, portability, readability and safety by placing restrictions on the ISO C++ language standard in order to limit the flexibility it allows.
www.codingstandard.com

JSF AV++ — the Joint Strike Fighter Air Vehicle C++ Coding Standard by Lockheed Martin, available for free, is intended to help programmers develop code that does not contain defects that could lead to catastrophic failures resulting in significant harm to individuals and/or equipment. (Not to be confused with the significant harm to individuals and/or equipment caused by military software that is working perfectly fine.)
www.jsf.mil/downloads/documents/JSF_AV_C++_Coding_Standards_Rev_C.doc

A long list of static analysis tools for many programming languages can be found here:
http://en.wikipedia.org/wiki/List_of_tools_for_static_code_analysis

Patriot Missile Bug (1991)

During the first Gulf War, a US Patriot missile system in Saudi Arabia failed to intercept an incoming Iraqi SCUD missile, leaving 28 soldiers dead and injuring around 100 other people. The cause was a rounding error in the time calculations done by the software making it ignore some of the incoming targets.

28 dead, 100 injured

software bugs kill people

platform-specific, unspecified, undefined, implementation-defined, or otherwise poorly defined behavior. Here are a few rules that prohibit the use of some popular functions and libraries:

- The error indicator `errno` shall not be used;
- The library <locale.h> and the `setlocale` function shall not be used;
- The signal handling facilities of <signal.h> shall not be used;
- The input/output library <stdio.h> shall not be used;
- The functions `atof`, `atoi` and `atol` from the library <stdlib.h> shall not be used;
- The functions `abort`, `exit`, `getenv` and `system` from the library <stdlib.h> shall not be used;
- The time handling functions of library <time.h> shall not be used.

Note the interdiction of the input/output library <stdio.h>. Yes, you are supposed to roll your own `printf` function.

In the end it is up to you

The example rules listed above—all taken from real coding standards—may seem severe, but they are not written in stone. For many rules coding standards allow exceptions. Other rules are debatable, and it is up to you to decide if you want to respect them or not. On the Internet you can find rather philosophical discussions about certain rules showing that some are even open for interpretation. Even a coding standard is subject to bugs.

(130271-I)

[1] GNU Coding Standard: http://www.gnu.org/prep/standards/standards.html

[2] Top 25 of most dangerous software errors: http://cwe.mitre.org/top25/

Deadly Radiation Therapy (1985-1987, 2000)

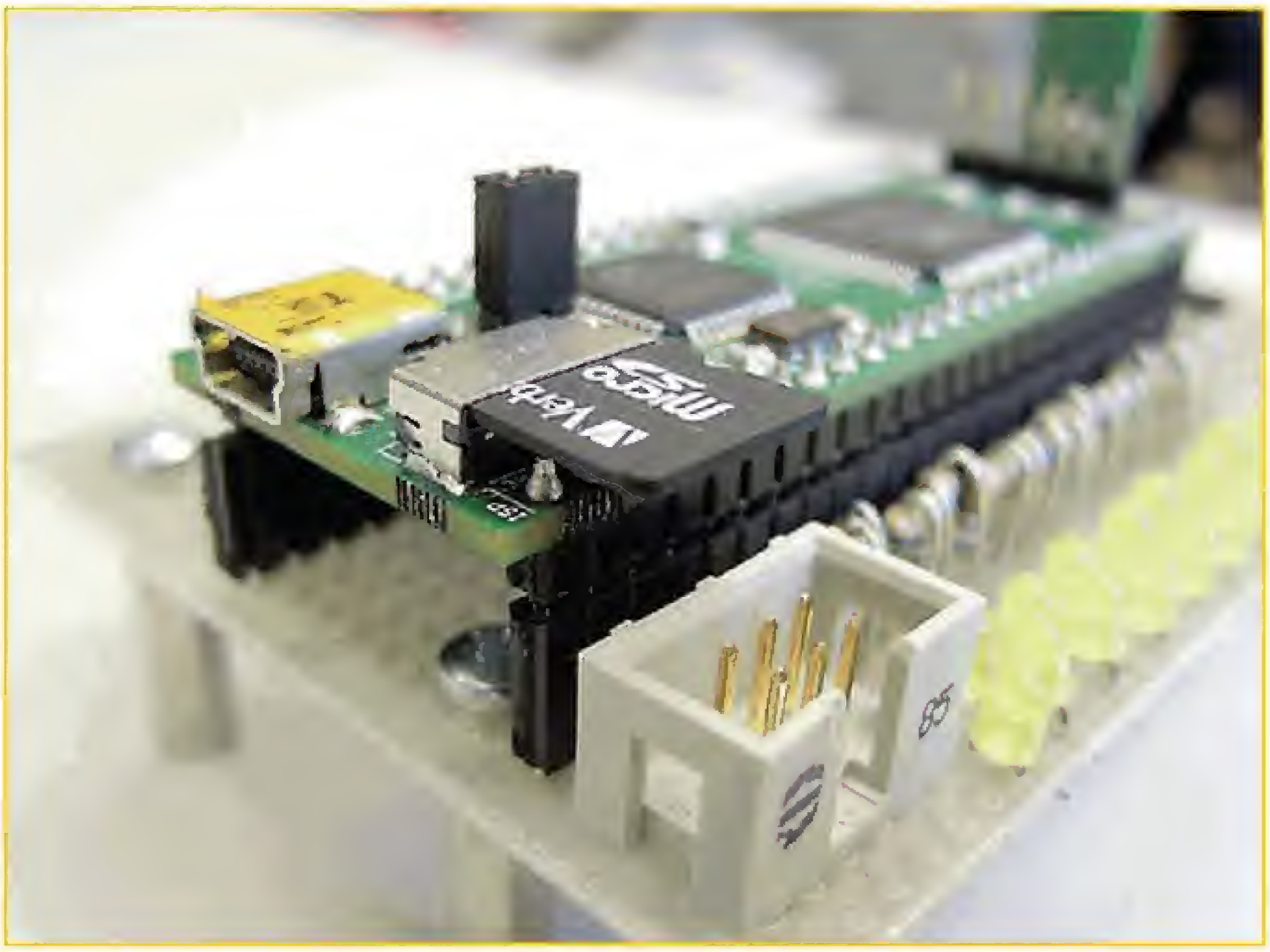
Due to buggy software the Therac-25 medical radiation therapy device could miscalculate the radiation doses it should administer. Some patients received up to 100 times the intended dose, which killed at least three of them. A similar bug surfaced in Panama City in 2000, where therapy planning software delivered different doses depending on the order in which data was entered. This bug killed at least five patients.

at least 5 dead, many injured

> There are two ways of constructing a software design: One way is to make it so simple that there are obviously no deficiencies and the other way is to make it so complicated that there are no obvious deficiencies.
>
> – C.A.R. Hoare, The 1980 ACM Turing Award Lecture


THE NFV EVENT
NOVEMBER 20-21, 2013 | SANTA CLARA, CA
TMC
SOFTWARE TELCO Congress™
Break Free of Hardware
Attention Carriers:
Plan Your Move to an NFV World
• Lower your costs using commoditized hardware
• Decrease time to market for new services
• Run production, test and reference facilities on the same infrastructure
• Leverage pure software companies to pursue new business models
• Allocate resources in near real-time
• Securely share hardware resources utilizing multi-tenancy
• Improve power management by switching off servers during non-peak hours
Diamond Sponsors:
Metaswitch Networks
ADTRAN
Alcatel·Lucent
Platinum Sponsors:
CISCO
GENBAND
sansay
saisei
Learn More at Software Telco Congress.
Register with Code ELEKTOR
and Save 15%
www.softwaretelco.com
#softwaretelco
@softwaretelco

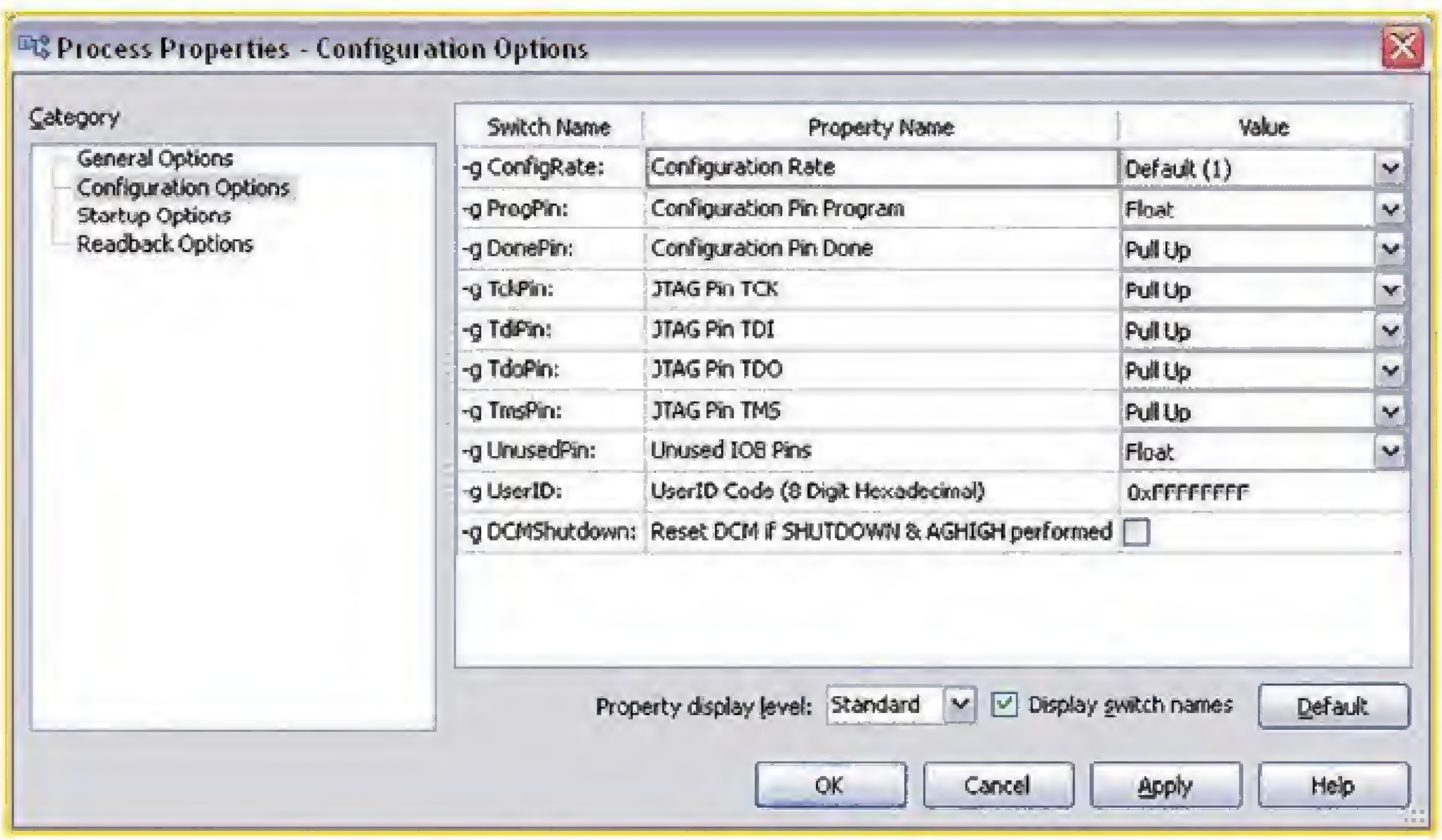
Keep the pins afloat



By **Thijs Beckers**
(Elektor Editorial)

The Elektor FPGA Development Board published in the December 2012 issue [1] uses a micro SD card to store the configuration data needed by the FPGA at startup. The card can also be used to store data, either by the FPGA, the microcontroller, or a PC when connected via USB.

When working on an application for the board, designer Raymond Vermeulen suddenly ran into troubles using the SD card. After programming the FPGA with freshly written firmware, the whole FPGA board wasn't detected by Windows anymore. Several attempts were made to fix this, like unplugging and reconnecting the USB cable, checking for program errors, et cetera, but to no avail. It seemed like the SD card had suddenly gone faulty. Testing the card in a reader however proved it to be functioning correctly, so the error had to be somewhere else. But where?



Every possible design flaw was checked and double checked, every solder joint tested, but everything turned out to be flawless. Retracing his steps, Raymond programmed the FPGA one last time and then it dawned on him: when programming the FPGA the software development environment requires a fair amount of parameters to be set. In the configuration options (see screenshot) unused I/O pins may be configured. In fact, they should. Exactly then Raymond admitted to having neglected just what he had been telling all users of the Elektor FPGA Development Board to do: set the unused I/O pins to *float*.

Since the data pins of the SD card are directly connected to both the microcontroller and the FPGA on the Elektor FPGA Development Board, it is mandatory these pins are left floating by the FPGA, when not in use. The development suite defaults unused I/O to *pull down*, which in most cases is fine. But in this application it is not, and this setting has to be actively changed to *float*!

After Raymond corrected this setting and reprogrammed the FPGA, the FPGA Dev Board came to life again and Raymond's application worked right off the bat. A clear case of *practice what you preach*, Raymond! Relieved he found the culprit he turned to me as I just entered the Labs and asked if anyone had an interesting story for this month's .LABS pages...

(130036)

Internet Link

[1] www.elektor.com/120099


USB Add USB to your next project. It's easier than you might think!
DLP-USB1232H: USB 2.0 UART/FIFO
HIGH-SPEED 480Mb/s
Only $28.95!
• Multipurpose: 7 interfaces
• Royalty-free, robust USB drivers
• No in-depth knowledge of USB required
• Standard 18-pin DIP interface; 0.6x1.26-inch footprint
DLP-IO8-G
8-Channel Data Acquisition
Only $29.95!
• 8 I/Os: Digital I/O
Analog In
Temperature
• USB Port Powered
• Single-Byte Commands
DLP-IOR4
4-Channel Relay Cable
DLP-TH1b
Temp/Humidity Cable
DLP-RFID1
HF RFID Reader/Writer
DLP-FPGA
USB-to-Xilinx FPGA Module
DLP Design
www.dlpdesign.com

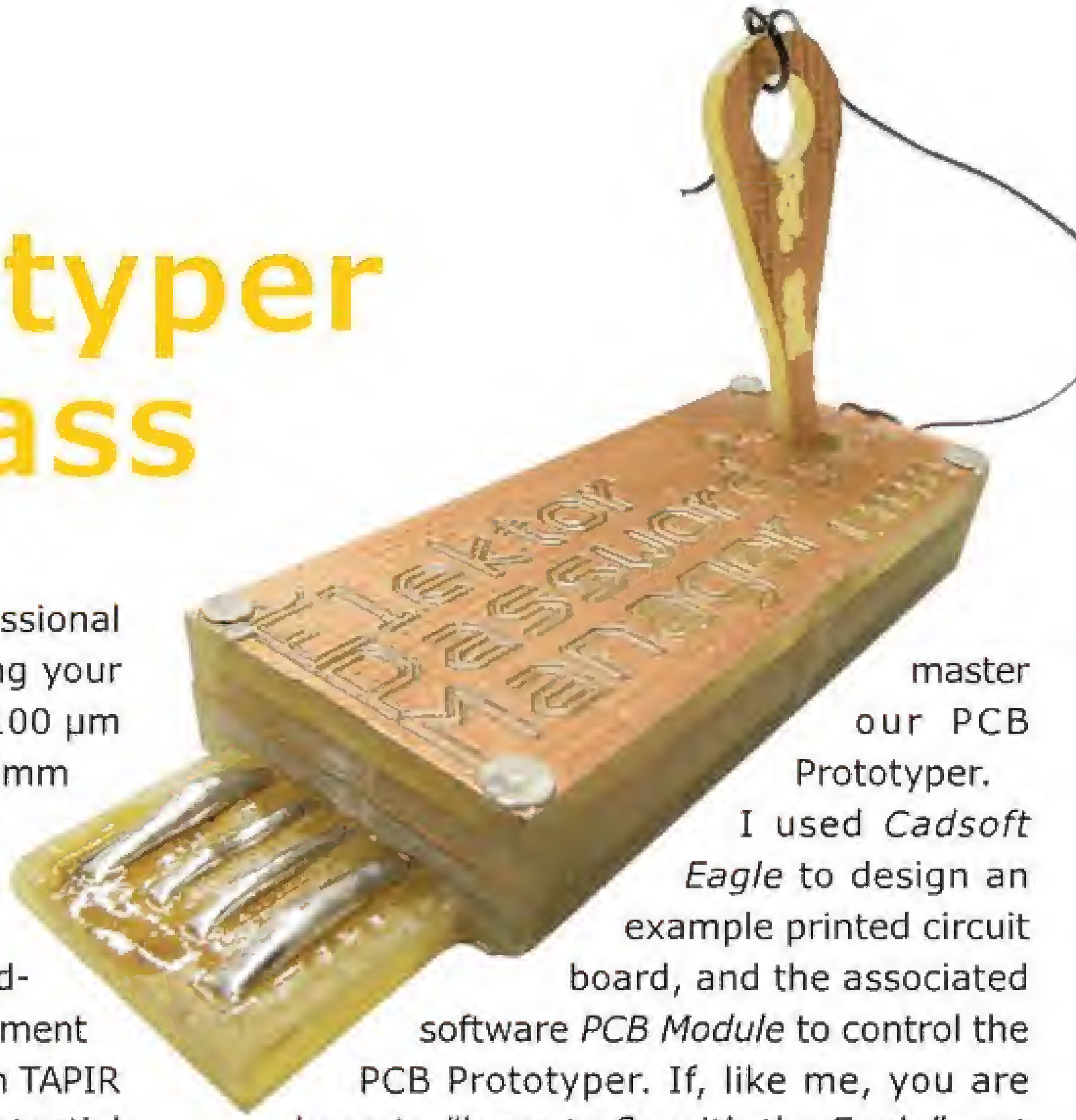

www.ezpcb.com
Professional PCB & PCBA Supplier
New Website Is Online!
Better Design
More Powerful
Easier to Use
EzPCB
Instant Quote & Pay
1 to 40 Layers
Prototype to Production
Amateur to Professional
order now
Prototype start at $10/ea
2L 4"x4" each
Free Shipping!
http://www.ezpcb.com


Take out a FREE membership to Elektor.POST
• The latest on electronics and information technology
• Videos, hints, tips, offers and more
• Exclusive bi-weekly project for GREEN and GOLD members
• Elektor behind the scenes
• In your email inbox each Friday
elektor post
The beginning of a beautiful friendship
Silicon based life forms of the World, unite!
Elektor TV goes Linux
elektor
Register today at www.elektor.com/newsletter


ELEKTOR
Preferred Suppliers
Tel. 1-978-281-7708
Fax 1-978-281-7706
Email ElektorUSA@smmarketing.us
INTEGRATED Ethernet PLCs for OEMs
Built-in Ethernet
MODBUS TCP/IP
Digital and Analog I/Os
PWM/PID/Stepper Control
From $119
TRI TRIANGLE RESEARCH INTERNATIONAL
Tel : 1 877 TRI-PLCS
web : www.tri-plc.com/ek.htm

PCB Prototyper Master Class

By **Aurélien Moulin**
(Elektor Labs Trainee)



The Elektor PCB Prototyper [1] is a professional PCB routing machine designed for routing your own PCBs with isolation tracks down to 100 µm width, and drilling holes as small as 0.2 mm (.008'') in diameter. Here at Elektor Labs—in the cellar to be accurate—we use our own PCB Prototyper machine when a PCB is needed in a hurry for building a prototype, or when we want to experiment with PCB shapes. Inspired by our very own TAPIR [2] we looked into the PCB Prototyper's potential for producing custom-shape PCBs. After some fiddling we were able to persuade the machine to produce the shapes we had in mind. To our great satisfaction we even mastered fabricating our own 'multilayer' PCB (see photo).

Eager to share our positive experience with you, our esteemed reader and member, allow me to guide you in pictures through the steps we took to master our PCB Prototyper. I used *Cadsoft Eagle* to design an example printed circuit board, and the associated software *PCB Module* to control the PCB Prototyper. If, like me, you are keen to "learn to fly with the Eagle", get the book! [3]

(130128)

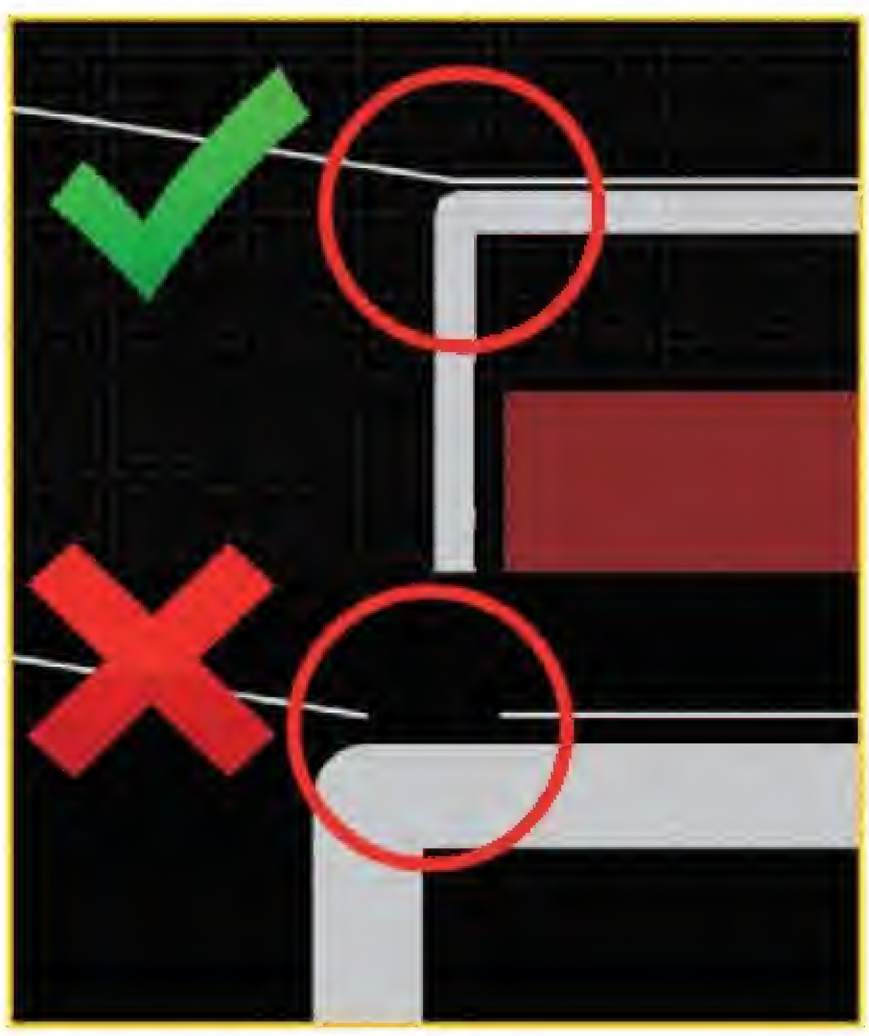
Internet Links

[1] www.elektor.com/PCBprototyper

[2] www.elektor.com/tapir

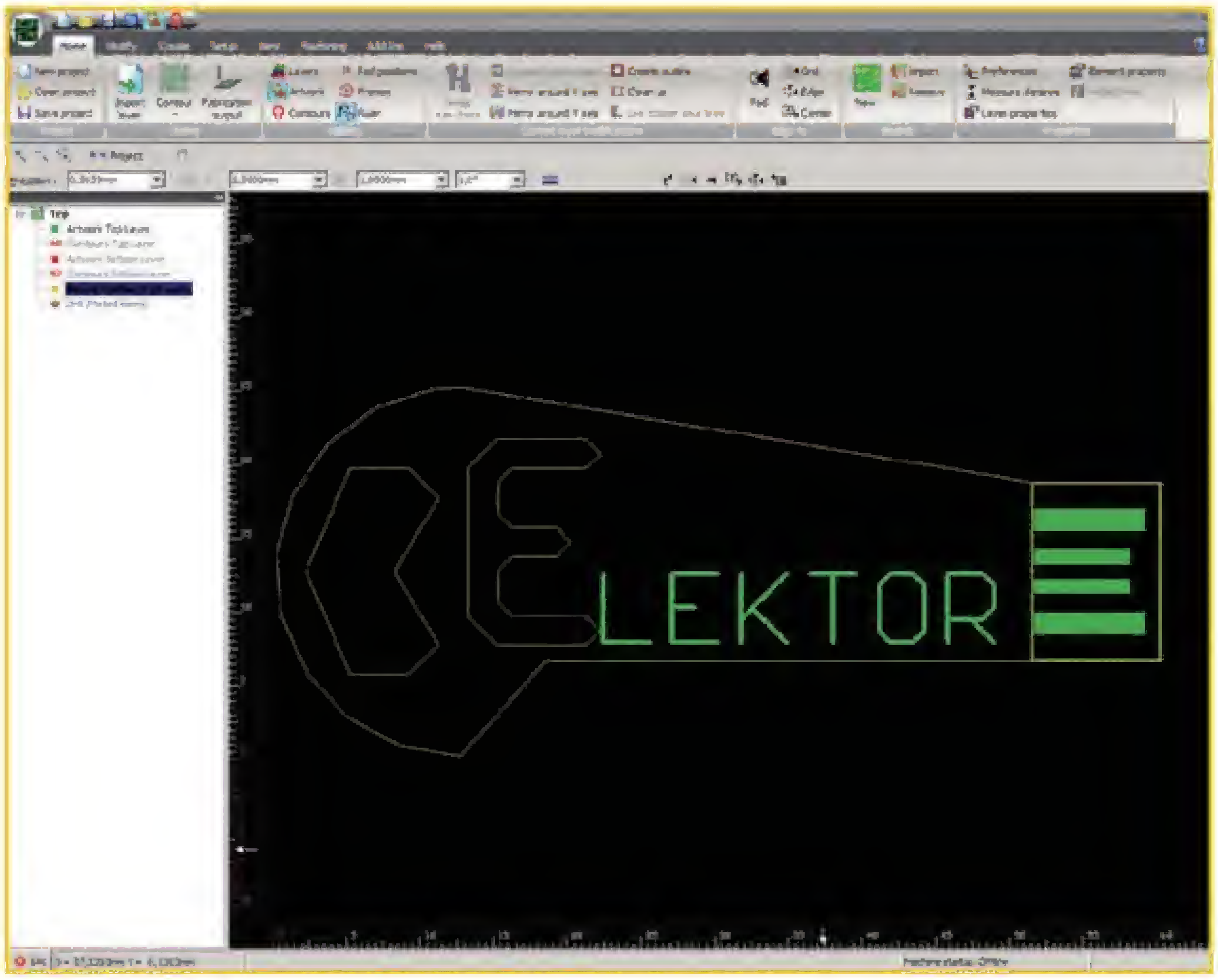
[3] www.elektor.com/eagle

1 In Eagle, select the *wire tool* to draw lines or the *arc tool* to produce round shapes. Make sure you select layer *20 Dimension* and set the width to 0. Now sketch up your design.

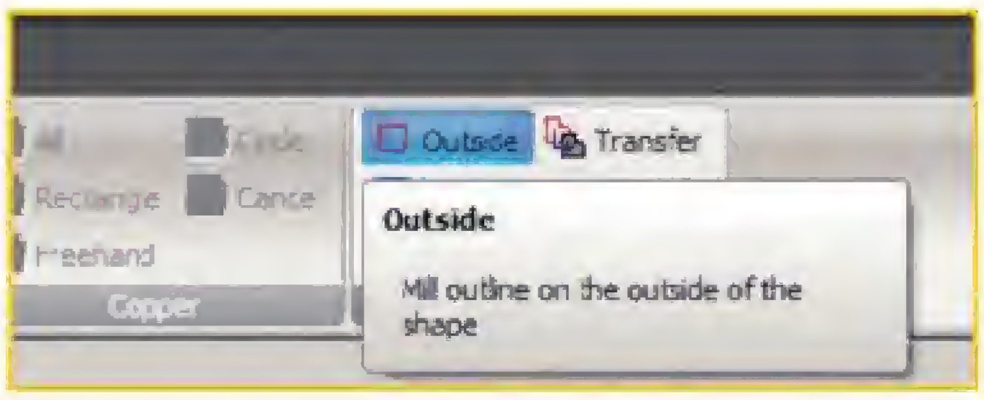


2 Verify that the lines form closed polygons, so that each shape delimits an inside and an outside area.

3 The green area is not part of the PCB, and has to be cut out. Bright red areas will be cut out from the inside of the PCB. Export your design using the CAM Processor. Select *GERBER_RS274X* and select the *Dimension* layer only. Click on *File* to save the output (name it 'dimension') and click *Process Job*. Repeat this for the top layer (select *Top*, *Pads*, and *Vias*; don't forget to rename your file before processing the job). To create the drill file, instead of *GERBER_RS274X*, select *EXCELLON* and repeat the process. This concludes the Eagle part.



4 Launch the *PCB Module* software and open a new project. Click *Import Layer*, select your top layer and import it into the top layer. Import a second layer and select your dimension file. Import this file into *Board Outline/Cut-outs*. Also import the drill file if you need it.



5 Now let's define the inlines and outlines (terms used in *PCB Module*). Click the *Modify* tab and select the *Outside* tool.

6 Put your pointer over the global shape. The color will change from yellow to white, indicating which polygon you're going to set as an outline. Clicking the outline should yield this.

7 Now select the *Inside* tool and apply it to the two shapes to be cut out from the PCB. Note that the 'E' is not properly filled. This calls for some manual fine-tuning. Cancel the last trace (*ctrl+z*).

8 You can try to correct this in *Eagle* (using a wireframe for the form of the character), but this means you have to redo all your CAM processing. Instead we came up with this quick & dirty fix: Draw a polygon (the tool hides behind the *Create* tab) that roughly follows the center of the character lines and set it to *inline* using the *Inside* tool.

9 Now we need to add some *Break-out tabs*. Use the *Break-out tabs* tool from the *Create* tab and click on a straight part of the outline. The software automatically centers the breakout. The default size is 2 mm (.08") which is fine, but you can adjust this to your liking. A proper end result should look like shown here. Your design is now ready for the usual processing stages (contour; tool list; machining). Happy routing!

WeatherPro with Sensor Support from Sensirion

The new version of WeatherPro for Android 3.0 now also supports sensors integrated in the cell phone, such as the humidity and temperature sensor from Sensirion. The sensor data is displayed in the daily forecast. This enables users to compare the data from their current location with the forecast values, making them better prepared for any weather.

David Kaiser, the man in charge of product management for WeatherPro at the MeteoGroup weather service, is pleased to be working with Sensirion: "Many devices, such as the Samsung Galaxy S4, have an integrated sensor from Sensirion. We wanted to make the data generated by the sensors available to our customers in those places where they usually go for their daily weather information. There's not always an official weather station right where an app user happens to be. The sensor data thereby provides interesting comparison values."

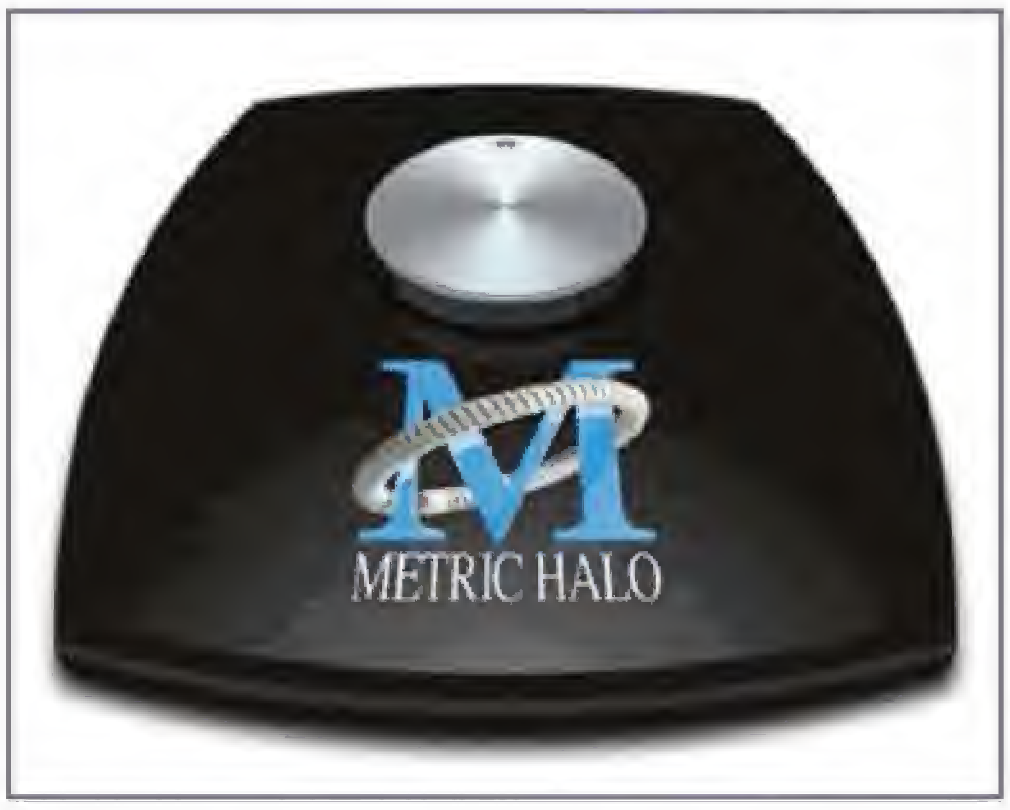
Andrea Wüest, Market Manager Mobile at Sensirion, believes that measuring the humidity and temperature of the immediate environment gives rise to many new opportunities. "Everyone now has their own individual weather station with them, which, thanks to WeatherPro, is also linked to the latest weather forecasts."

The use of such a sensor is only possible thanks to innovative technology in both the hardware and software. The hardware is the smallest humidity and temperature sensor currently available. It was specially designed for mobile end-devices by the Swiss high-tech company Sensirion and optimized to fulfill the industry's unique requirements. The company not only provides the sensor, but also the corresponding software. Without it, the heat generated by the smartphone itself would make any measurement of the ambient temperature impossible. Sensirion is currently the only company to offer an all-in-one solution for ambient sensing of humidity and temperature for smartphones.

www.weatherpro.eu www.smart.sensirion.com (130202-I)

MIO Console 5.6

Metric Halo announces the immediate availability of MIO Console v.5.6, a free software upgrade for all users of the Mobile I/O family of audio interfaces, including the award-winning 2882, ULN-2, LIO-8 and ULN-8. Version5.6 includes the following new features: I/O inserts for accessing external hardware from within the MIO Mixer, ConsoleSync hardware/software synchronization technology, AAX ConsoleConnect plug-in for compatibility with Pro Tools 11, saving of system boot states and support for EuCon 3.0. In addition to these new features, v.5.6 continues to improve stability and compatibility with current and future versions of Mac OS X.



ConsoleSync is a unique enhancement to the Mobile I/O family that benefits both new and experienced users. ConsoleSync allows MIO Console to read the complete state of any attached hardware seamlessly, automatically and without any disruption of running audio. ConsoleSync auto loads the mixer configuration, complex signal processing chains, Monitor Controller settings, analog I/O configuration and even window layout from the hardware.

With ConsoleSync, new users will experience a dramatically reduced learning curve for accessing the power of MIO Console. Experienced users will value the ability to have MIO Console re-connect to the hardware with exact recall of the current hardware state and no disruption of audio. For live sound and monitoring applications this allows the user to disconnect and re-connect the computer or quit and launch MIO Console without being concerned about introducing dropouts to ongoing primary and backup recordings or to the monitoring paths for talent.

I/O inserts streamline the use of external analog and digital processors within the MIO mixer. This new feature also allows inserting processors that are hosted on the computer directly within the signal flow of the MIO low-latency hardware mixer. This enables the use of host-based reverbs and delays with greatly simplified routing. This free upgrade continues the Metric Halo tradition of adding value to the Mobile I/O platform as well as enhancing the product for new users. A 30-day money-back guarantee backs Metric Halo's hardware products so you can try them out in your studio with no risk.

www.mhlabs.com (130306-VII)

Library of Apps for Jupiter Turn-Key DSP

Symetrix announces the addition of seventeen new apps for its family of Jupiter app-based turn-key DSPs. The three different hardware units in the series—Jupiter 4, Jupiter 8, and Jupiter 12—differ only in their input/output counts and are user-controllable from Symetrix ARC wall panel remotes, third-party systems, and Symetrix ARC-WEB browser-based software for smartphones. The addition brings the total number of available Jupiter apps close to one hundred and expands the already impressive range of situations for which the Symetrix Jupiter provides reliable processing at an extremely competitive price point.

Six core apps form the heart of the addition and are elaborated according to the input/output counts of the three Jupiter hardware devices. "BGM Zone Mixer 1" provides background music routing to multiple zones with two levels of priority. "Dual Matrix Mixer 1" provides flexible mixing and routing and allows integrators to assign any input to a mix, assign those mixes to submixes, and route submixes to outputs. "Gain-Sharing Automixer 3" provides a gain-sharing automixer with feedback processing on the outputs together with matrixed outputs. Similarly, "Gating Automixer 3" provides a gating automixer with feedback processing on the outputs together with matrixed outputs. "Priority Zone Mixer 2" provides a multi-zone priority mixer with paging and SPL computing. Finally, "Sound Reinforcement 12" provides heavy input processing and full-range outputs with feedback processing.

The growing library of Jupiter apps are optimized for specific applications and venues including houses of worship, auditoriums, retail and hospitality establishments, sports facilities, and transportation terminals. The time needed to learn the software is zip. With Jupiter turn-key DSP sound contractors end up doing what they do best, which is to dial in really great sound.

www.symetrix.com (130202-III)

Quad-Core CPU in Fanless Chassis

Logic Supply has released the LGX AU970 Intel® Core™ Fanless Computer. Designed for demanding industrial applications like high-end automation, data collection and surveillance, this tough, wide-temperature system takes ruggedized, high-performance computing to a whole new level.

The AU970 offers high-performance computing for even the most graphics-intensive applications. It supports up to 16 GB dual-channel 1333 DDR3 memory, and comes equipped with a dual- or quad-core third-generation (Ivy Bridge mobile) Intel Core i3/i5/i7

CPU and QM77 chipset.

Third-generation Intel Core processors efficiently deliver top-of-the-line graphics performance while taking up very little space. Logic Supply is among the first to offer this technology in a fanless chassis.

A comprehensive array of I/O allows for a wide range of connectivity options, including two eSATA and CFast ports for external storage, a SIM card slot for 3G connectivity in remote deployments, and an 8-bit Digital I/O port. The AU970 has 3 LAN ports (one with Intel iAMT for remote support), 10 USB ports, 4 COM ports and 2 DVI ports for dual independent display. The system has a secure terminal block connector for wide input 9-32V DC power with remote power-switch capability.

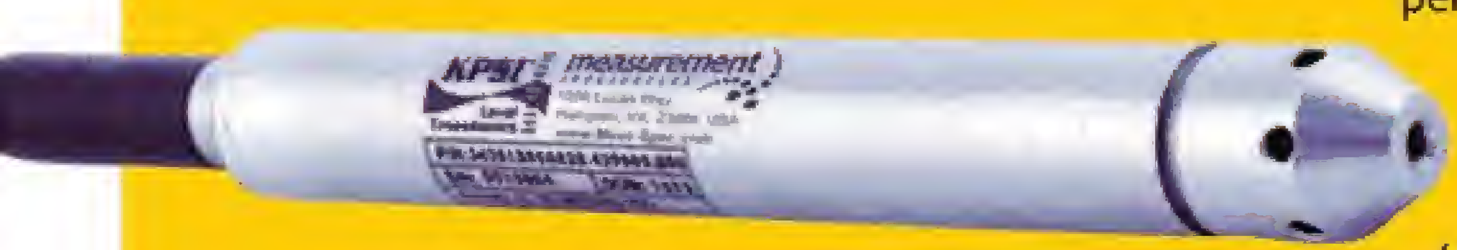
The system's heavy-duty heatsink and chassis provide fanless cooling, with a wide operating temperature of –20C to 55 °C. The fanless design ensures silent operation, as well as reliability in highly dust- and dirt-prone environments. It also comes equipped with a wall-mounting kit for easy installation.

The AU970 is available exclusively from Logic Supply. Also available is the LGX AU972 Expandable Computer, which, in addition to the above, offers dual PCI express expansion slots for a variety of additional I/O—from network port additions to COM ports and specialized hard drive hot-swap bays.

www.logicsupply.com (130306-VIII)

New Temperature Range for KPSI 342

Measurement Specialties has improved the operating temperature range for the KPSI Model 342 submersible level transducer. The 342 transducer can now withstand temperatures up to 85°C for use in hostile fluids providing a 4-20mA analog output with digital transducer performance.



The Model 342 is a small bore (3/4" body diameter) submersible hydrostatic level transducer that combines Measurement Specialties' well known sensor competencies with the latest in Application Specific Integrated Circuit (ASIC) technology. The addition of the ASIC has given the Model 342 digital performance in an analog product over the entire operating range.

The Total Error Band specification (±0.25% FS) over the compensated temperature range (–20 to 85°C) eliminates the user having to combine multiple performance specifications to realize the total accuracy of the transducer. While numerically larger, it is a more accurate specification because it defines linearity of the sensors' performance as a total rather than using the traditional definition, which takes the difference between the largest positive and negative linearity values. The Model 342 is available in vented, sealed, and absolute formats to 300psi (700' H_2O WC level) full scale output.

www.meas-spec.com (130202-II)

True RMS Self-Powered AC Voltmeter Fits 30.5 mm / 1.20 Inch Round Panel Cutouts

Murata's Type DMR20-1-ACV "nanometer" self-powered four-digit bright LED voltmeter can measure the true RMS value of its input from 85 to 264 VAC to within 0.1 V resolution. Designed to be completely self-contained, the low cost two-wire voltmeter requires no additional components or connections apart from the AC voltage it is measuring. It is capable of accurately measuring quasi-sine AC source such as modified, modified 2-step and modified 3-step sine wave in addition to conventional sine, triangle and square wave inputs.

The voltmeter fits an industry standard "oiltight" 30 and 30.5 mm / 1.2 inch round panel cutout. The 7.6 mm

/ 0.3 inch four-digit LED display is housed in a rugged round polycarbonate case that provides protection against dust, moisture, vibration and shock. It is supplied with an EPDM rubber gasket and plastic hex nut that aid protection to IP67 / NEMA6 specification for water ingress.

The DMR20-1-ACV is ideal for measuring the AC line voltage of a wide range of applications such as primary line power, power distribution units and backup power generation sources.
The voltmeter consumes a maximum of 30 mA when used at 250 VAC / 60 Hz.
Round knockout punch tooling can be ordered with the DMR20. Free sample / evaluation units for qualified OEMs are available from stock.

www.murata-ps.com (130202-IV)

PC Oscilloscopes with Deep Buffer Memory and USB 3.0 Superspeed Interface

With up to 500 MHz bandwidth on four channels, and an industry-leading 2 G samples of buffer memory, the new PicoScope 6000 Series has both the performance and the advanced analysis capability to speed debug of today's complex electronic designs.
The PicoScope 6000 Series employs hardware acceleration and a USB 3.0 interface to acquire and display many megasamples of data per screen update without slowing down. Engineers can observe large portions of their design's electrical behavior at one time, and in great detail, which helps to reduce debug cycles and enables electronic design projects to be completed on schedule.
As Alan Tong, Managing Director of Pico Technology, explained, "The PicoScope 6000 Series is the highest-performance USB oscilloscope, with deeper buffer memory as standard than any other oscilloscope, and is capable of detailed circuit analysis. We have provided a suite of advanced debugging tools, included as standard with the scopes, so that engineers who are developing complex electronic systems will find all the functions they need."
All models include an integrated function generator or arbitrary waveform generator (AWG), advanced triggering, automatic measurements with statistics, an FFT spectrum analysis mode, comprehensive waveform maths, mask limit testing, and serial decoding for popular industry standards such as I²C, SPI, UART, CAN, LIN and FlexRay.
The PicoScope 6000 Series scopes are compact and portable devices that fit easily in a briefcase, and include a five-year warranty as standard. Prices start at €2414 / $3292 / £1995 for the 250 MHz model with function generator, through to €5439 / $7417 / £4495 for the 500 MHz model with arbitrary waveform generator and 2 G samples of buffer memory. A set of four high-quality matched probes is supplied with every scope.

www.picotech.com (130202-V)

Forze VI:
A Hydrogen-Powered Racecar

By **Tessel Renzenbrink**
(Elektor TTF Editor)

The Forze VI, one of the first hydrogen-powered racecars, was unveiled on September 9 at the disused Valkenburg airfield near Katwijk (Netherlands). All components of the car were built by a team of 70 students at the Delft University of Technology.

The Forze VI and The Team

The symbolic value of the location was a good match for the objective of the Forze team. In scarcely less than a century, aviation technology has radically changed our world and now allows us to travel to the other side of the globe in less than 24 hours. With their hydrogen-powered racecar, the students hope to contribute to another technology-driven revolution, this time in the realm of clean energy.

The Forze VI weighs just under 2,000 lbs. (880 kilograms), achieves a top speed of 138 mph (220 km/h), and accelerates from 0 to 60 mph (100 km/h) in 4 seconds. The heart of the racecar is the fuel cell system, where hydrogen reacts with oxygen to produce electricity and water. The generated electrical power drives two electric motors, each with a rated power of 190 kW (260 HP). The car can race for 30 minutes at top speed on the fuel in its two tanks, which together hold three kilograms of hydrogen at a pressure of 350 bar.

Sustainability can be cool

In Hanger 2, Edgar van Os—the founder of the Forze team—explained to the audience what motivates the students to devote so much time (sometimes up to 80 hours per week) to the project. "We want to show that sustainable energy can also be cool. Sustainability is always presented in a negative sense: switch off the light, turn down the heating, and so on. Our approach of combining sustainability with racing was radical and pioneering. That is why we also gave a lot of attention to the appearance of the car. It is not a nerdy engineering-student vehicle, since that doesn't interest the general public."

Van Os set up the Forze H2 team in 2007 entirely on his own initiative, without any help from sponsors or professors. The goal was to build a go-cart that could participate in the Formula Zero competition in 2008. This first series of races for vehicles powered by hydrogen fuel cells was initiated to foster zero-emission technology "from the source to the wheel". The Forze I vehicle competed against entries from five other university teams and came out on top.

Since then the Forze platform has undergone a number of evolutionary developments, with each team building on the knowledge of its predecessors. Development of a hydrogen fuel cell designed and built by the team, to replace the ready-made fuel cells used previously, started in 2010. This was installed for the first time in the Forze IV, which also marked the change from a go-kart platform to a small racecar. Now the team has delivered the first full-size race car in the form of the Forze VI.

Hydrogen

After the introductory remarks by Van Os, Dr. Bernard Dam, a professor of chemistry at Technical University Delft, gave a short talk on the benefits of hydrogen fuel systems. "Global warming is a scientific fact. It is therefore necessary to reduce CO_2 emissions by 80 to 90 percent by 2050, relative to the level in 1990. Mobility plays a major role in this. Reducing emissions in the industrial sector is difficult, which is why the answer must be found in the transport sector and the urban environment—office buildings, houses, shops and so on."

Right now there is a lot of interest in the development of electric vehicles equipped with batteries. If the electricity is generated from renewable energy sources, this results in zero-emission vehicles. A side benefit is that this pool of vehicles can act as a distributed storage system. Wind energy and solar energy deliver a fluctuating yield of electricity, which makes it necessary to create facilities for storing energy.

However, according to the professor "it is by no means certain that affordable batteries that allow vehicles to travel a thousand kilometers will become available in the future." The popularity of electric cars still suffers from "range anxiety": the fear of getting stuck alongside the road somewhere with an empty battery. A hybrid hydrogen-powered car can provide a remedy by using the fuel cell as a range extender.

Dr. Dam and his colleagues are working on a method to produce low-cost, sustainable hydrogen in the future. Presently most hydrogen is obtained from natural gas. This process still generates CO_2 emissions, and natural gas is not a renewable resource. The method being investigated by Dr. Dam is called water splitting. This involves first immersing a photo-electrochemical cell in water and then exposing it to sunlight. The resulting chemical reaction splits the water into its constituent elements: hydrogen and oxygen. The researchers recently achieved a milestone by attaining an efficiency of 4.9%. This means that nearly 5% of the solar energy is converted into hydrogen. They expect to be able to achieve their efficiency target of 10% within three years, which would make the technology commercially viable.

Water

When hydrogen reacts with oxygen in the fuel cell, the only by-product is pure water. The Forze VI produces a liter of water every minute. The water can be stored in the car or discharged onto the road surface. Neither solution is ideal: the first option makes the car heavier, while the second option leads to jeers from other motorists. For this reason, the Delft team came up with an innovative alternative. They use the water to cool the brake system, which causes the water to evaporate. The kinetic energy from braking is also converted into electrical energy and fed back into the system.

The Forze VI from the student team is a full-fledged racecar that will compete with cars powered by gasoline. It will be entered in 25 events during the 2013–2014 race season. One of the highlights is an attempt to break the lap speed record for electric vehicles at the Zandvoort track. The record is currently held by the Tesla Roadster. At the famous German Nürburgring race circuit, the Forze team hopes to enter the record books as the builders of the fastest hydrogen fuel cell powered car that ever rode on the Nordschleife (North Loop).

(130203-I)

Freystedt's Audio-Frequency Spectrometer (1935)

Restoration of a landmark in electro-acoustic measurement technology

By Dr. Götz Corinth
(Germany)

As opposed to time consuming mechanical, optical and graphic/mathematical methods, electronic engineering in the mid-1930s brought new possibilities to sound analysis. Filter circuits allow individual frequencies and frequency bands to be stressed or suppressed. The 'search frequency' method allows highest resolution to be achieved at constant absolute bandwidth, but at the expense of analysis speed.

A pass-band filter fitted ahead of a multichannel looping oscillograph (in German: *Schleifenoszillograph*) enabled parallel recording of multiple frequency ranges. Attempts at real-time visual assessment remained problematic though.

In 1934, Erich Freystedt at Siemens & Halske's Central Laboratory improved the process by means of 27 parallel third-octave filters at the input, covering the 30-18,000 Hz human hearing range ("analysis with constant relative band-

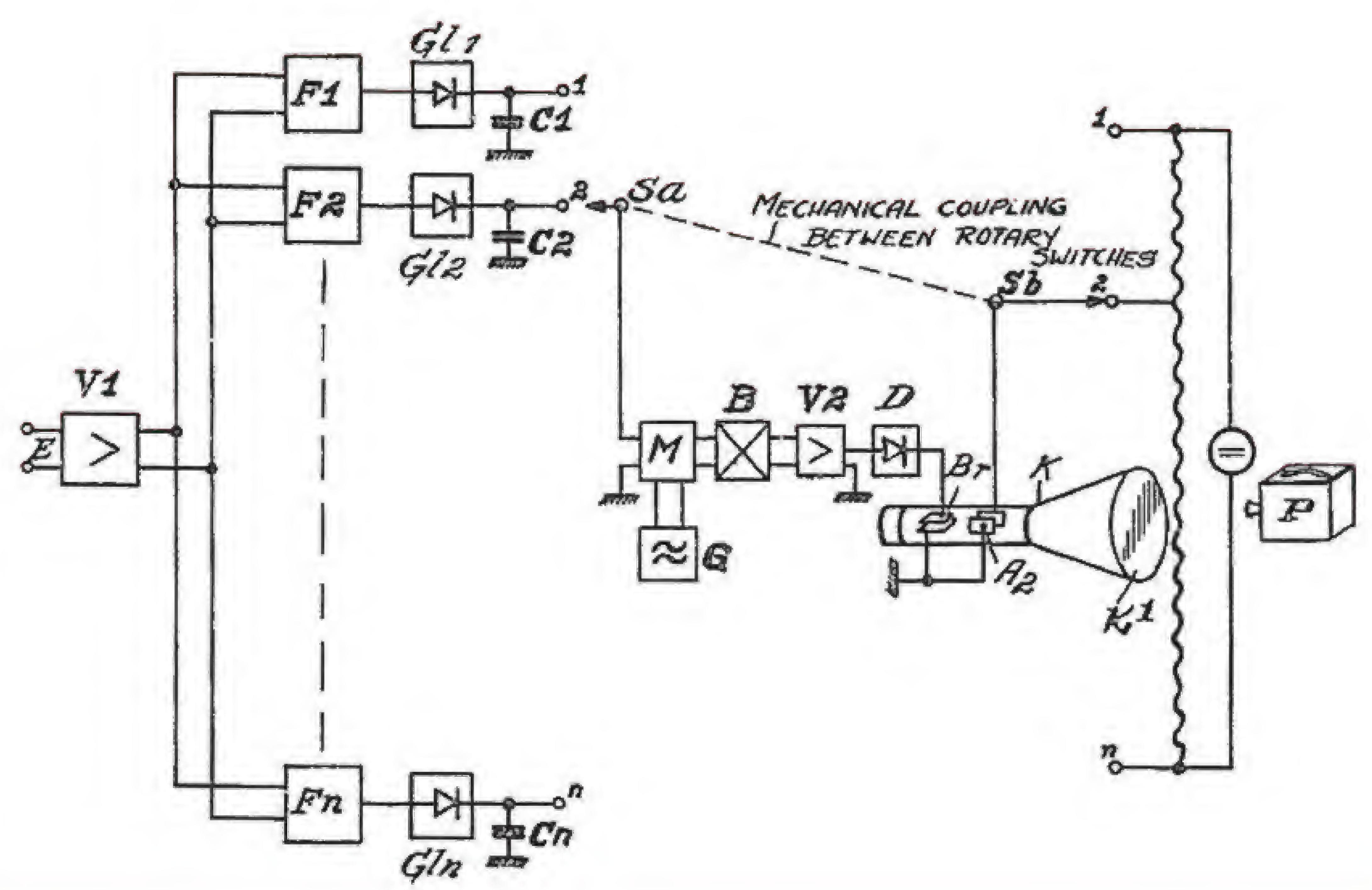


Figure 1.
Block diagram (partial) of the Audio-Frequency Spectrometer. Note the photo camera, 'P'. US Patent Office file # 2,159,790, May 1939; *Z. Tech. Phys. 16 (1935)*, p. 294 (Freystedt)

width"). Looking at **Figure 1**, after rectification and short-term storage in capacitors, the filter output signals are continuously sampled by a motor-driven switch assembly, and subsequently applied to the vertical deflection plates of a cathode ray tube (CRT), on which the signal levels appear as columns.

Synchronously, an appropriate DC voltage level is applied to the horizontal deflection plates. The resulting image on the oscillograph represents a line spectrum. Each line represents the peak voltage obtained from "third-octave" filtering, carefully taking into account the electrical properties of capacitors and filters (**Figure 2**). Depending on the sampling rate a sort-of continuously updated image of the spectrum is generated, of which the time domain can be registered additionally.

Where is the Spectrometer?

There was the vague, but not entirely unfounded hope that such an instrument might just be part of a private collection in Rüsselsheim, Germany. Rüsselsheim? Errm... Opel? (later, General Motors Europe, *Ed.*) What! Noise tests on car engines? Indeed, the suspicion was confirmed: this spectrometer was built in 1937 and appeared to be at least fairly well preserved, at least on the outside. All 200 lbs. (85 kg) worth of AF Spectrometer was hauled into the author's electronics laboratory with difficulty.

EST^D 2004

Retronics is a monthly section covering vintage electronics including legendary Elektor designs. Contributions, suggestions and requests are welcome; please telegraph editor@elektor.com

Where is the documentation?

In terms of documentation, initially there were only two original publications from Siemens and ATM Scientific Publications. Later, a Dutch collector was contacted through PTB's (Physikalisch-Technische Bundesanstalt) Technical Acoustical Division. He was willing & able to supply the original documentation from Siemens. The papers also contained an equipment price indication from 1938: RM (Reichsmark) 5,500. According to Germany's Federal Statistical Office that's $35,000 or €25,000 referenced to the year 2000.

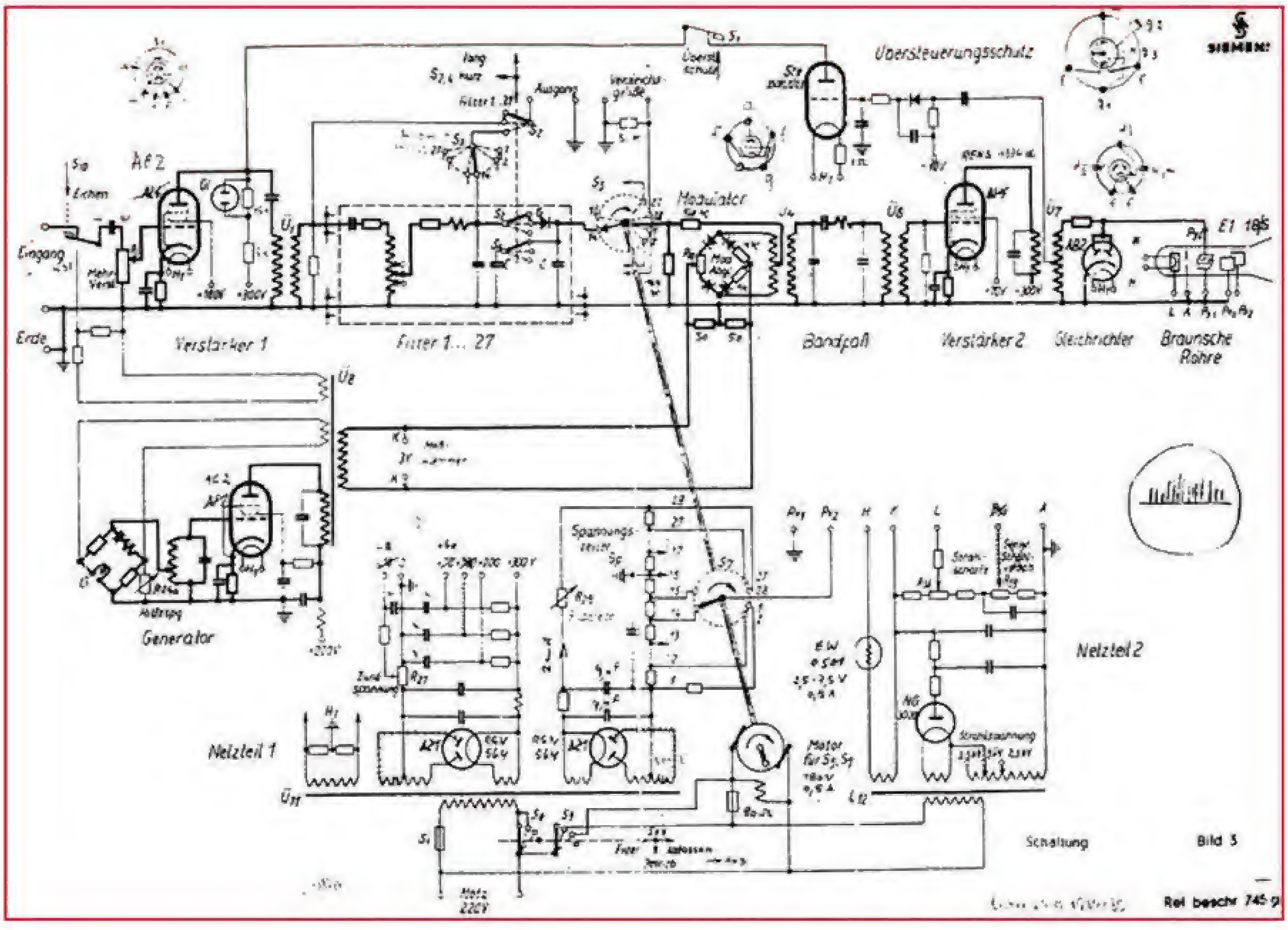


Figure 2.
Detailed schematic of the Spectroscope instrument (from factory documentation *Rel beschr. 745 g.* (Siemens & Halske company, 1936). Schwere Wörter translated: *Eingang* = Input; *Verstärker* = Amplifier; *Übersteuerungsschutz* = Overdrive Protection; *Bandpaß* = Bandpass; *Gleichrichter* = Rectifier; *Braunsche Röhre* = Cathode Ray Tube; *Netzteil* = Power Supply; *Schaltung* = Circuit.

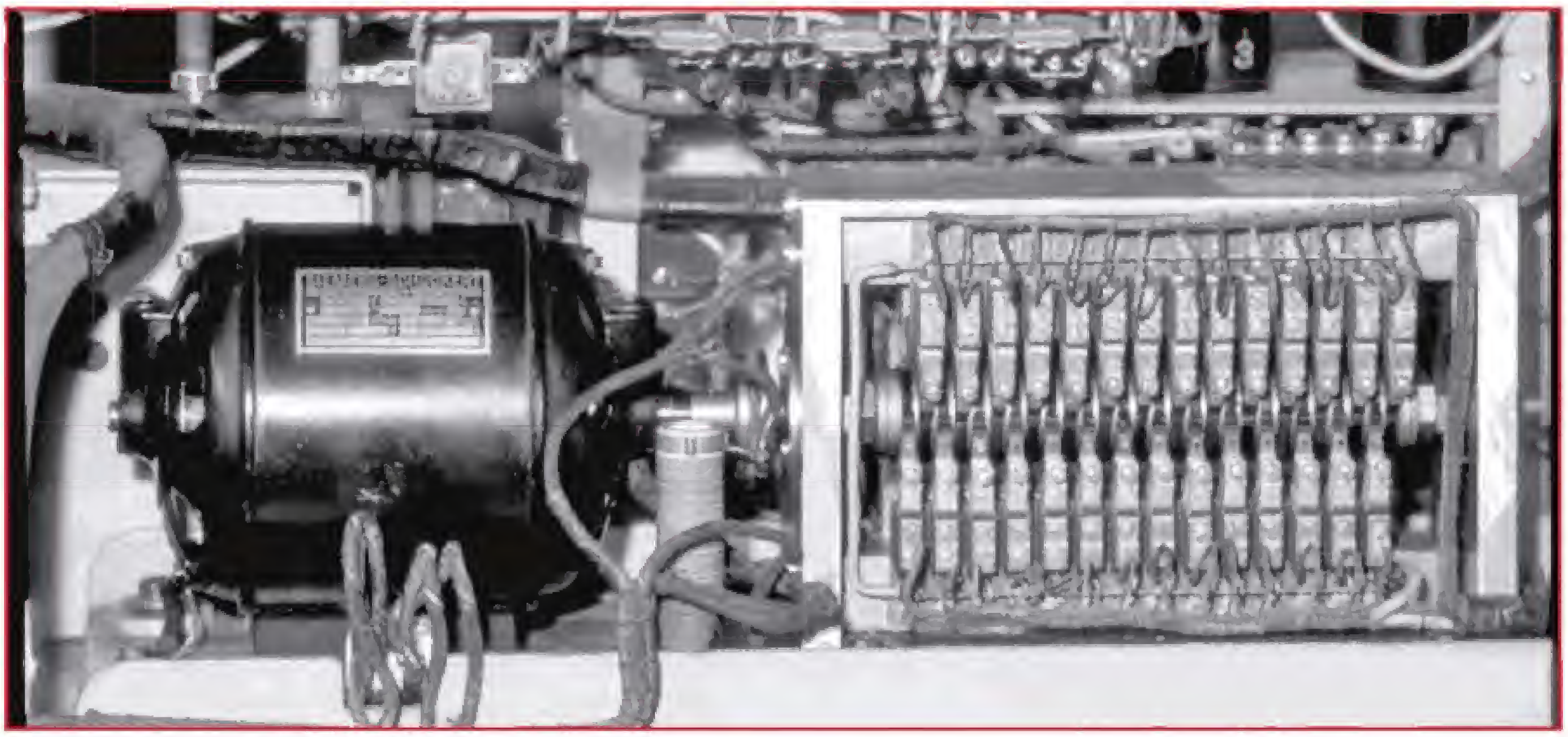


Figure 3.
Drive motor and camshaft switching gear comprising 54 spring sets. (original photo)

Electrical checking, investigating, restoring

First cautious attempts: the drive motor for the two sets of 27 cam switches (**Figure 3**) was stuck; no high tension (HT) for the 7-inch (18 cm) cathode ray tube ("Braun'sche Röhre") with asymmetrical deflection. Unfortunately, this tube had become unusable due to corrosion of the wires at the pinched base. Only the audio frequency (AF) part of the instrument seemed to be functional to some extent.

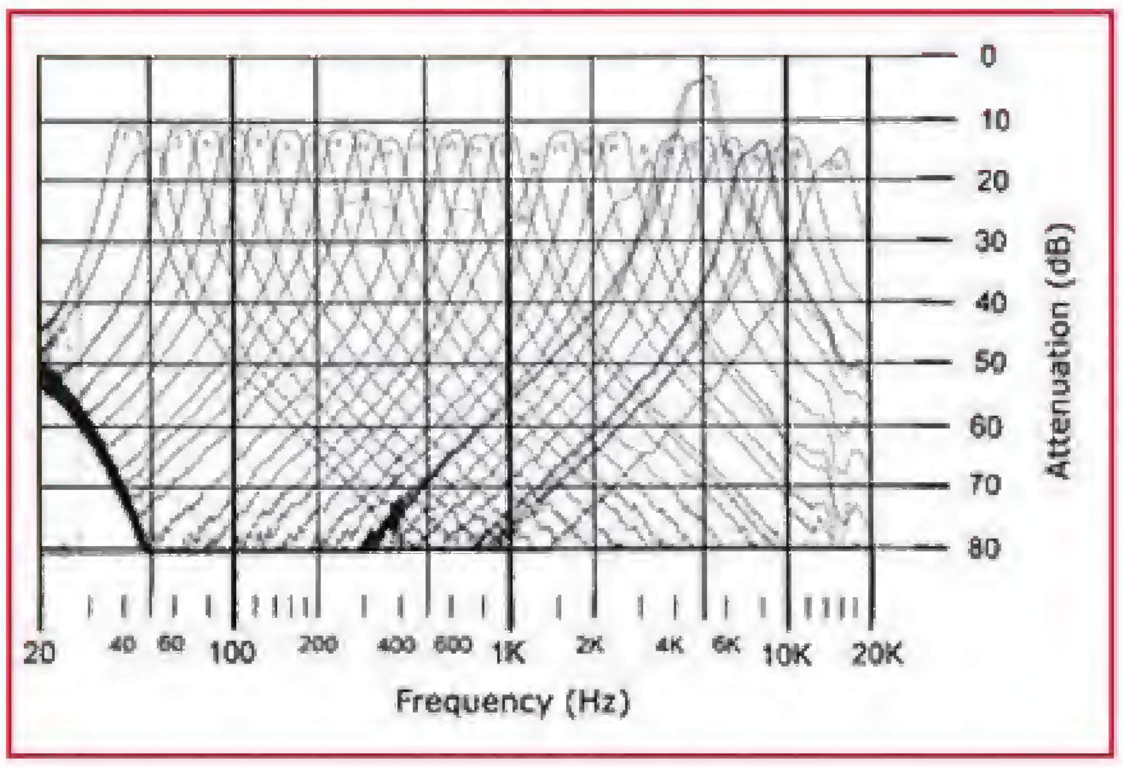


Figure 4.
Frequency response of the filter before the equipment restoration. (original recording)

The high tension transformer had to be rewound twice until it kept working in the circuit. A specialist firm in Holland overhauled the cathode ray tube, giving it a completely new electron gun and screen. The AF section of the instrument was thoroughly checked, and unsafe components got replaced.

The readjustment of many filters was necessary as well as labor intensive (**Figure 4**). Here the capacitors showed up increased capacitance—probably due to shrinkage of insulating paper over a period of nearly 70 years—ruling out the simple knack of connecting a few small extra capacitances in parallel. Two capacitor decade banks and a computer-controlled sweep generator ("wobbulator") allowed the target pass-band curves to be set up again (**Figure 5**). The "Sirutor" cuprite (Cu_2O; a minor ore of copper) rectifiers between the filter and the storage capacitors all proved completely intact.

To be able to operate the "new CRT in a vintage envelope" the setting options for the tube's aux-

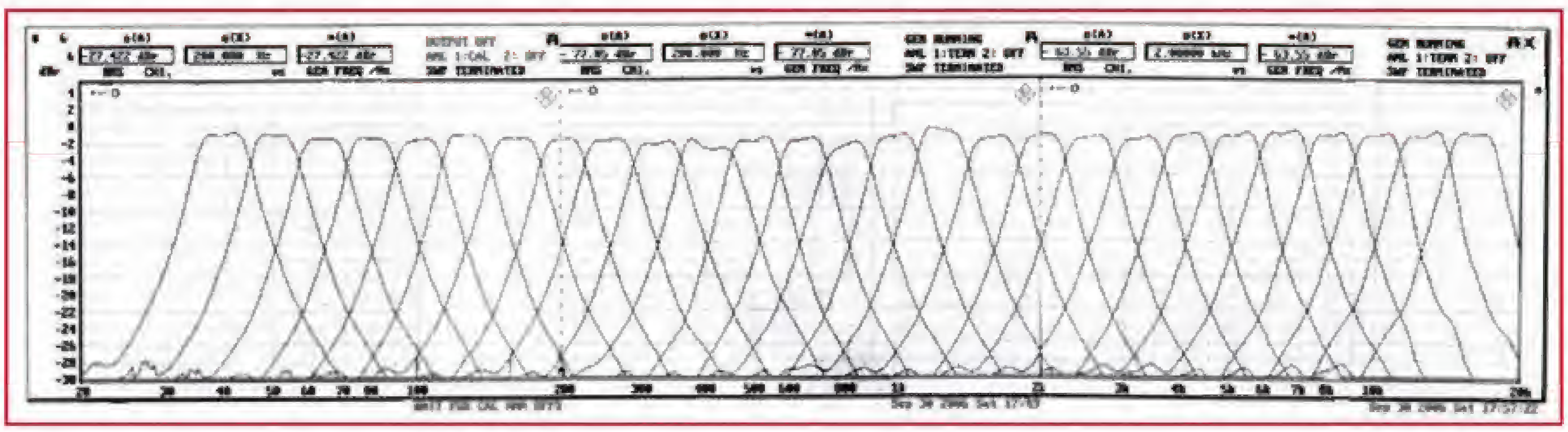


Figure 5.
Frequency response of the filter after recalibration. (original recording)

iliary voltages had to be reconsidered, resulting in practice in a small circuit board being etched and corresponding potentiometers installed on it, for retro-fitting inside the instrument. To avoid making excessive alterations to a historical piece of equipment, the strictly correct method was relinquished of applying symmetrical voltages to the CRT by a push-pull driver circuit. CRT trapezoidal skew and astigmatism proved tolerable at not too high deflections (**Figure 6**).

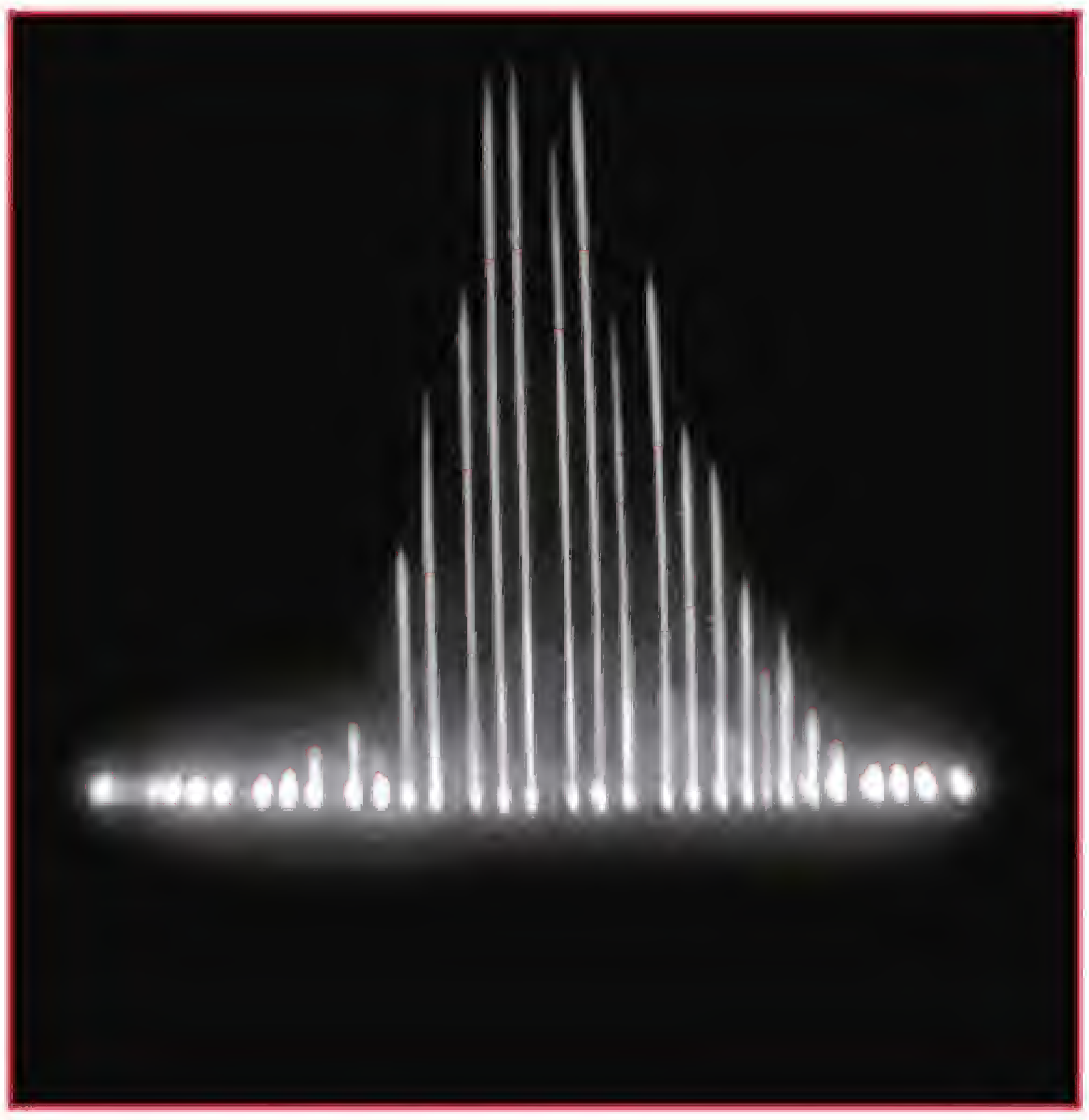


Figure 6. Screen snapshot of a distorted sinewave voltage. (original recording)

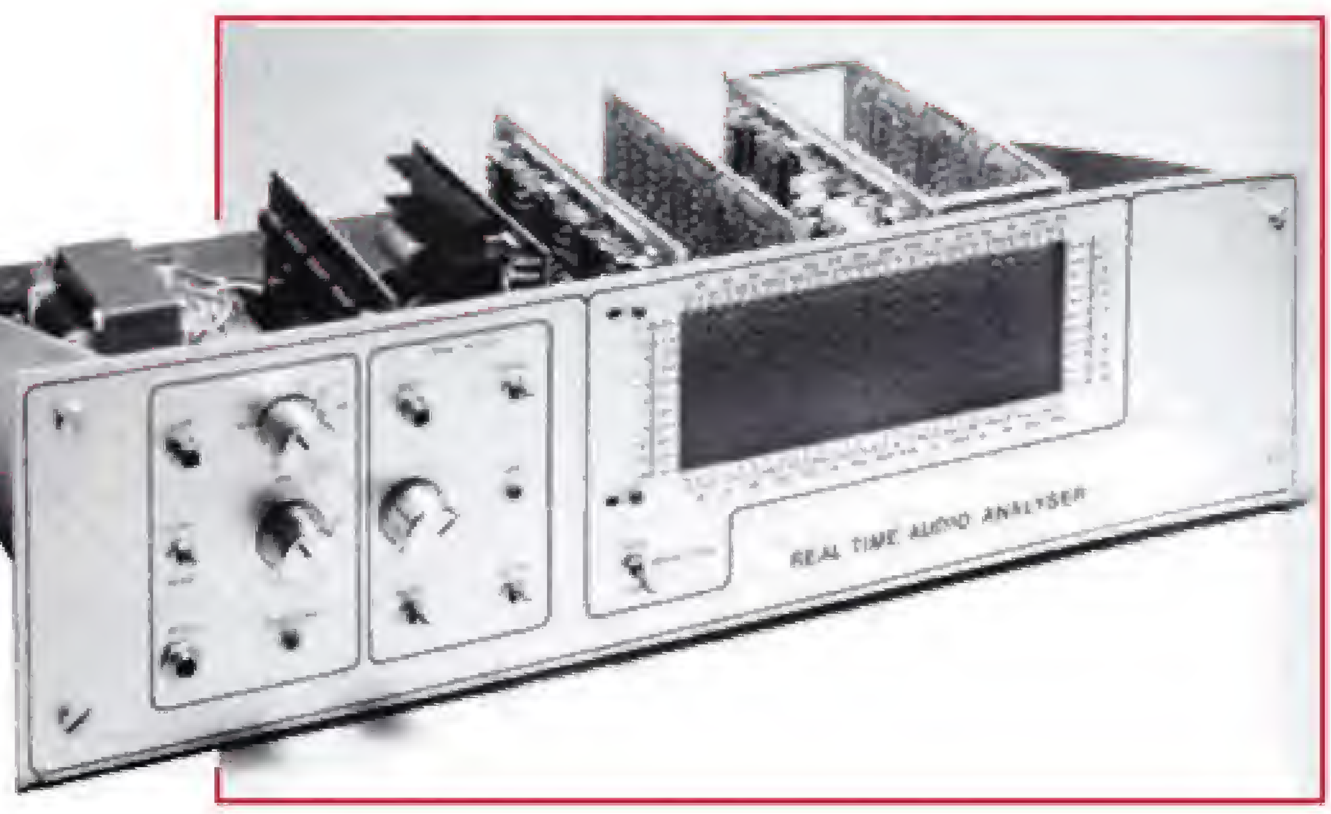
Mechanical work

The motor fitted with new bearing bushes (home made on a precision lathe), and all 54 spring assemblies adjusted using a stroboscope, a spectrum appeared for the first time on an external oscilloscope.

After the restoration of the electrical parts, the exterior was rebuilt by repainting the cabinet, nickel-plating metal components and making new lettering on and around the controls (**Figure 7**).

Figure 7. View of the audio-frequency spectrometer near completion of the restoration work. The casing is removed to show the 27-section filter bank and the motor-driven contact array. The modern multi-color parts were required to electronically match the reworked CRT to the original circuit. (original photo)

Figure 8. Finished prototype of the Elektor real-time analyser, photographed from an Elektor May 1984 page. (original reproduction by copyright holder)



Back Home

The result of all the restoration work covering many weekends finally ensured that a monumental piece of electro-acoustic measurement technology was operational again. The finished instrument was discussed by the author in a presentation to the 2007 German Society for Acoustics Congress, and afterwards consigned as a free, permanent loan into the historical collection of the original manufacturer, Siemens, Munich.

Elektorized in 1984

The March, April and May 1984 editions of Elektor magazine (priced 95p/$3) contained a 3-part article series describing a "real-time analyzer" (*sic*), which may be considered the All-Solid-State & Performance-Plus version of the 1935 Siemens boatanchor described here, although that instrument failed to get a mention in 1984. In good 1980s fashion the Elektor DIY project contains vast quantities of discrete parts (how about 250-odd precision resistors), half a dozen of densely stuffed circuit boards mounted on a base board, a hefty power supply and a colorful readout comprising 330 LEDs. A pink noise generator is also included.

The lab prototype of the 1984 real-time analyser probably ended up in a dumpster in preparation of Elektor's office move from Beek to Elektor House, Limbricht, back in 2008. All we have today are the original articles. Not to worry— thirty years on we cheerfully run or the likes of it on a laptop PC. It's just a 500 KB download away. Some of you have ARM cores, DSPs and hidden speaker systems turning a bubble car or an Opel Kadett A into a Ferrari as far as sound is concerned.

Your Editor can't resist signing off with **Figure 8**— it's his best attempt at showing what the Elektor 1984 instrument looked like. It was designed by Harry Baggen who is still on the Elektor Netherlands editorial staff. Owners of an Elektor 1984 real-time analyser, please telegraph editor@elektor.com.

(130204)

Hexadoku Puzzle with an electronic touch

Here you stumble upon a new specimen of our popular Hexadoku, the 16-by-16 puzzle that's sure to keep you busy for a couple of hours. Hexadoku is not just for electronic engineers, but also for their relatives and friends! Find the solution in the gray boxes, submit it to us online, and you automatically enter the prize draw for one of four vouchers.

The Hexadoku puzzle employs numbers in the hexadecimal range 0 through F. In the diagram composed of 16 × 16 boxes, enter numbers such that **all** hexadecimal numbers 0 through F (that's 0-9 and A-F) occur once only in each row, once in each column and in each of the 4×4 boxes (marked by the thicker black lines). A number of clues are given in the puzzle and these determine the start situation.

Correct entries received enter a prize draw. All you need to do is send us **the numbers in the gray boxes**.

Solve Hexadoku and win!

Correct solutions received from the entire Elektor readership automatically enter a prize draw for one Eurocircuits PCB voucher worth **$140.00 (£80.00)** and three Elektor book vouchers worth **$60.00 (£40.00)** each, which should encourage all Elektor readers to participate.

Participate!

Before December 1, 2013,

supply your personal details and the solution (the numbers in the gray boxes) to the web form at

www.elektor.com/hexadoku

Prize winners

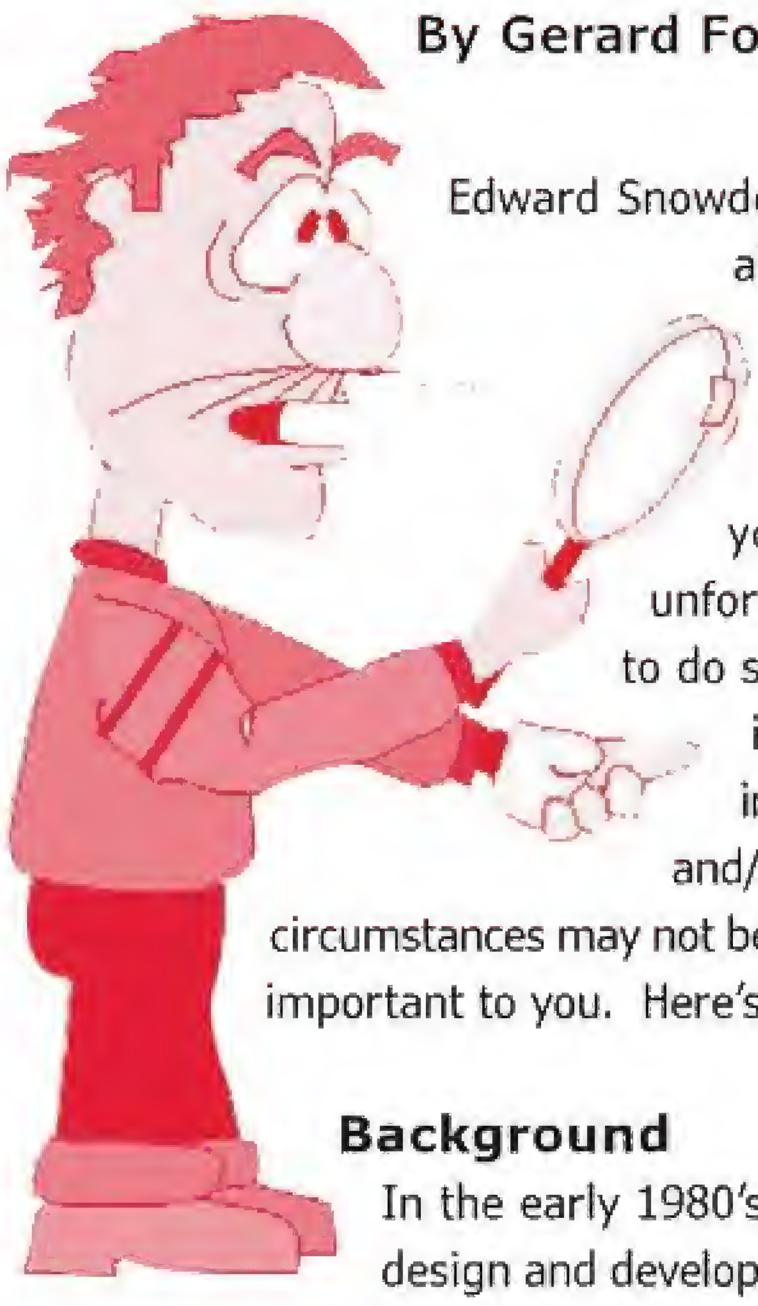
The solution of the September 2013 Hexadoku is: **569E8**. The Eurocircuits $140.00 (£80.00) voucher has been awarded to Emil Cugini (Switzerland). The Elektor $60.00(£40.00) book vouchers have been awarded to Arno Habermann (Netherlands), Richard Fleischmann (USA), and Arun Annaji (India).

Congratulations everyone!

		B				0		6				7	A	3	F
		9			F	7	1		B			8			2
			7		4	B					3		0		9
				2			3		8	7		4		1	E
				B			2	E	C	9	6		F		
	3	F		C	7		0	4			8	A			
				9							D	1		4	
	8	A	C					0		1	2				B
7		C	8		0		9			F	1	B		5	
F				7	1								9	2	D
	9			E		3	B			A			4	F	
A		2	4		C	5			D	8	9	0			
B		4	9	0			7	A					C		
			D	1			E	8		0				7	5
					8			1		6					
1			0	5		F	A								

5	6	9	E	8	1	B	F	4	2	C	0	A	7	D	3
8	F	4	A	9	2	5	C	D	3	7	E	B	0	6	1
B	C	0	7	6	A	D	3	5	F	1	8	9	2	4	E
1	2	D	3	4	7	0	E	6	9	A	B	8	F	C	5
2	E	B	8	7	F	3	4	C	0	6	9	D	1	5	A
3	5	F	D	A	6	9	0	7	1	4	2	C	8	E	B
6	0	1	4	B	5	C	D	E	8	F	A	2	3	7	9
7	9	A	C	2	8	E	1	B	D	3	5	F	4	0	6
F	7	2	0	5	9	1	8	A	B	D	4	E	6	3	C
4	1	5	B	C	D	6	7	8	E	2	3	0	9	A	F
A	D	3	6	E	B	F	2	9	7	0	C	1	5	8	4
C	8	E	9	3	0	4	A	1	6	5	F	7	B	2	D
9	4	6	F	D	C	7	5	2	A	8	1	3	E	B	0
D	3	7	2	F	E	A	B	0	5	9	6	4	C	1	8
E	A	C	1	0	3	8	6	F	4	B	7	5	D	9	2
0	B	8	5	1	4	2	9	3	C	E	D	6	A	F	7

Conscientious Objector



By Gerard Fonte (USA)

Edward Snowden has received a lot of publicity about revealing the CIA's analysis of telephone and internet activity in the US. So, it seems appropriate to discuss what to do when you are in such a situation. It is unfortunate but true that being asked to do something that you find unethical is not uncommon in the engineering arena (especially for government and/or product design). And while your circumstances may not be as conspicuous, they are certainly important to you. Here's my story.

Background

In the early 1980's I was working for an aerospace design and development firm that specialized in government contracts. I was chosen as lead engineer for a Flight Inspection project. Flight Inspection is the calibration of civilian airport equipment that identifies the position of aircraft as they near the airport. Obviously, it's very important that when the airport instruments say that the aircraft is ten feet to the right of the runway centerline, the aircraft is really and truly ten feet right of the centerline. This was a plum and very visible assignment and it was very important to me for my advancement at the company.

Unfortunately, some ignorant VP had already sold a "design" to the client. The idea was to put a digital TV camera in an aircraft; fly the aircraft fifty feet over the runway; take a picture of the runway number; analyze the picture to determine the aircraft's precise position over the runway; and then compare the "known" position with the airport's instruments for error. The best digital camera of the time had a resolution of about 1000 by 1000 pixels. There were eight-bit, single-board computers operating at a few MHz available. There were no real operating systems or math packages or optical recognition software. It all had to be written from scratch, in-house.

I had recently taken some graduate courses in pattern recognition and artificial intelligence and knew immediately that this was not an easy thing to do. I asked how precise the positioning of the test aircraft had to be. I was told: "To the inch". I swallowed hard and said nothing. I took the specifications to my desk and got busy.

You Can't Get There From Here

It was immediately apparent that there were major troubles. The runway markings were spray-painted by hand. I suspected that they could be off by at least an inch or two whenever the runway was re-painted—which was often. That was over the error budget by itself. (It was later determined that this error could be as much as five inches.) However, I ignored that for the time being and worked out a hardware error figure.

With 1024 pixels and a runway 150 feet wide, each pixel was 1.75 inches. The basic resolution of the camera was simply not good enough by nearly a factor of two. I then added in the speed of the plane, shutter speed of the camera (30 fps), altitude error (from a laser altimeter) and other error sources. The expected error was about 12 to 14 inches or over an order of magnitude worse than what was required. This did not include software error or runway marking error. I had a problem.

I talked to my immediate boss about this and he passed it upwards. After a week with no response, I put my error analysis on paper and passed that up the chain. After another week, I re-worked the analysis with more details and sent that paper through channels. The next week a different engineer was given the task to determine what the error was. His expected measurement error was two to three feet. However about half of that could be calibrated out, leaving an uncorrected error of about 12 to 18 inches.

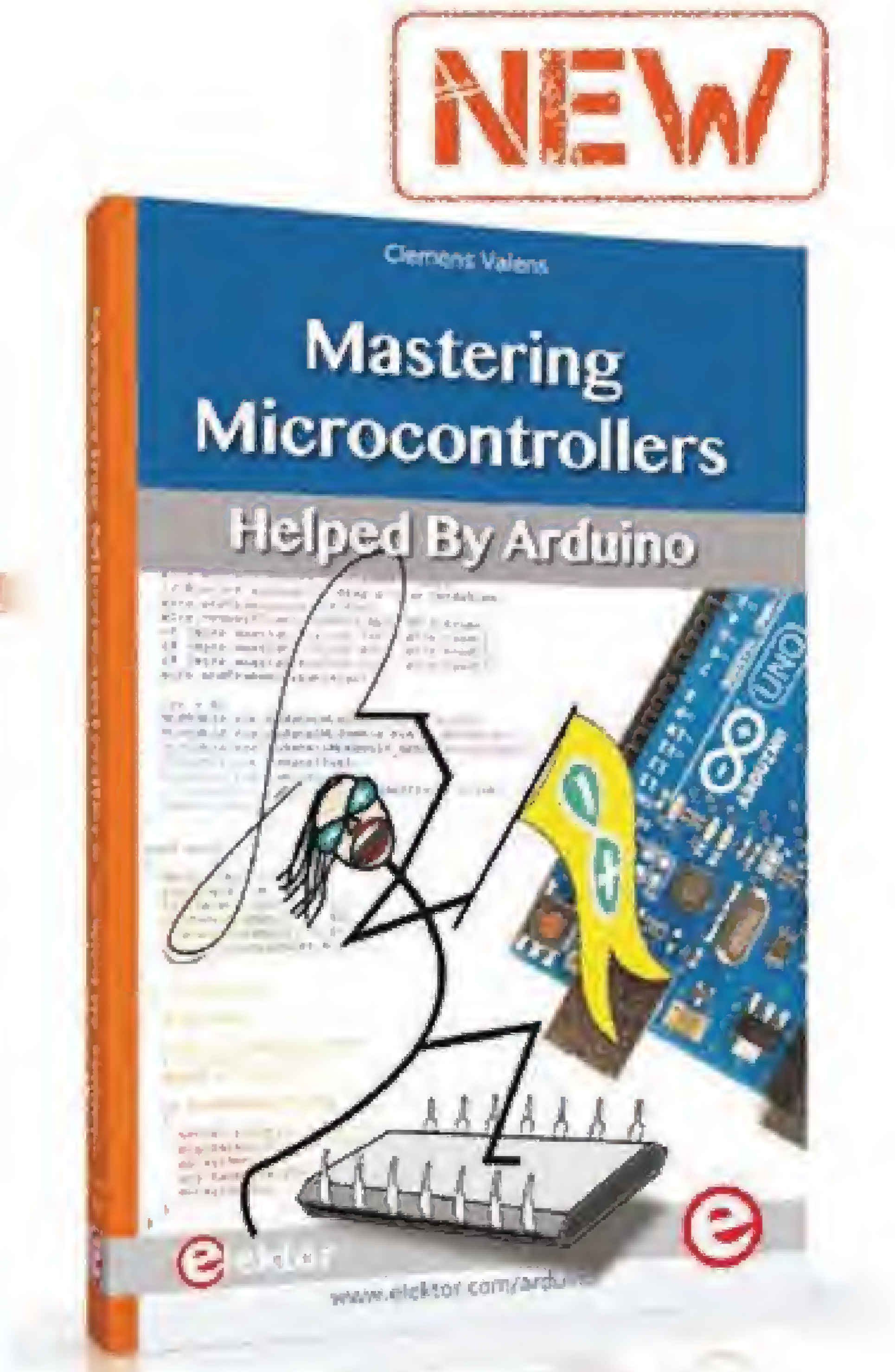
I thought on the problem and realized that they might not know any other method. So I came up with my own approach. It used several semiconductor lasers set at special angles on the plane with four bicycle reflectors glued to the runway lights. It was very cheap, very precise (0.1" resolution) and very simple. I built a demonstration model using my personal He-Ne laser and some lenses. I showed it to my boss, who liked the idea. However, a few days later I got the word that the idea was rejected because "we can't put anything on the runway." Not because of any legal or safety concern, but because the original VP's "design" said so.

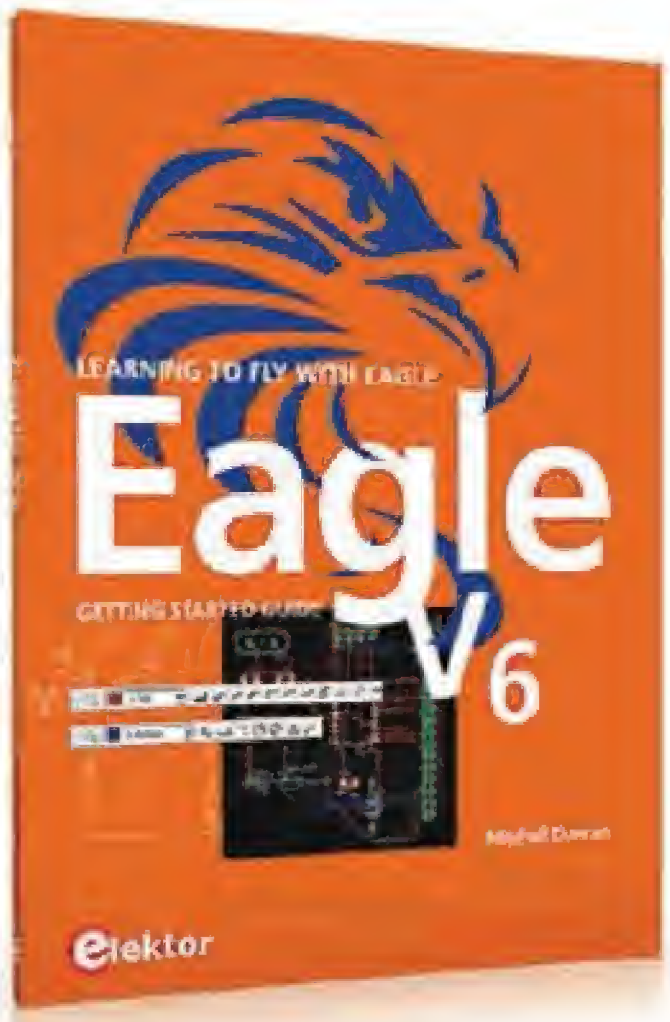
Final Answer

Several days after this I was in an informal meeting with my boss and a director. When the topic came up I asked: "How are going to address the error problem?' To this point I had gotten no feedback at all. The director, obviously annoyed and frustrated with my persistence, blurted out: "We'll cheat. Everyone does." I resigned a few weeks later. The project went forward with a special optical design placed on the ground, to one side of the runway. I don't know if the error problem was ever fixed. A couple of years later some of these same people were indicted on contract procurement irregularities.

Different people would make different choices. There is no "right' answer to this problem. Each individual has to evaluate the situation and choose the course that is best for him or her. However, I still think that I addressed the matter in a reasonable and proper manner. And that is the moral of this story.

(130372)

1


2


3


Limited Time Offer for GREEN and GOLD Members!
15% DISCOUNT
www.elektor.com/arduinobook

4


5


Helped By Arduino

1 Mastering Microcontrollers

The aim of this book is not only to let you enter the World of Arduino, but also to help you emerge victorious and continue your microcontroller programming learning experience by yourself. In this book theory is put into practice on an Arduino board using the Arduino programming environment.

Having completed this fun and playful course, you will be able to program any microcontroller, tackling and mastering I/O, memory, interrupts, communication (serial, I²C, SPI, 1-wire, SMBus), A/D converter, and more. This book will be your first book about microcontrollers with a happy ending!

348 pages • ISBN 978-1-907920-23-3
£34.95 • € 39.95 • US $56.40

Learning to fly with Eagle

2 EAGLE V6 Getting Started Guide

This book is intended for anyone who wants an introduction to the capabilities of the CadSoft's EAGLE PCB design software package. This book will quickly allow you to obtain an overview of the main modules of EAGLE: the schematic editor; layout editor and autorouter in one single interface. You will apply your knowledge of EAGLE commands to a small project, learn more about some of the advanced concepts of EAGLE and its capabilities and understand how EAGLE relates to the stages of PCB manufacture. After reading this book while practicing some of the examples, and completing the projects, you should feel confident about taking on more challenging endeavors.

208 pages • ISBN 978-1-907920-20-2
£29.50 • € 34.50 • US $47.60

The luxury of precision within everyone's reach

3 500 ppm LCR Meter

The remarkable precision of this device and its amazing ease of use are the result of careful design. It works so well behind its uncluttered front panel that one could almost forget the subtleties of the measurement techniques employed. A dream opportunity for our readers who are passionate about measurement to enjoy themselves. If, like us, you wonder at the marvels modern techniques bring within our reach, come along and feel the tiny fraction of a volt.

Set: main board and LCD board, assembled and tested
Art.# 110758-93
See www.elektor.com/lcrmeter

Display, buttons, real time clock and more

4 Elektor Linux Board Extension

This extension board was developed to further propel our Embedded Linux series of articles and the matching GNUblin board. It has a display, buttons, a real time clock and 16 GPIOs. Linux devotees, switch on your solder irons. The Linux extension board includes everything needed to provide the user interface for a wide variety of projects!

Module, SMD-populated and tested board, incl. LCD1, X1, K1-K4, BZ1, BT1 for home assembly
Art.# 120596-91
£31.10 • € 34.95 • US $50.20

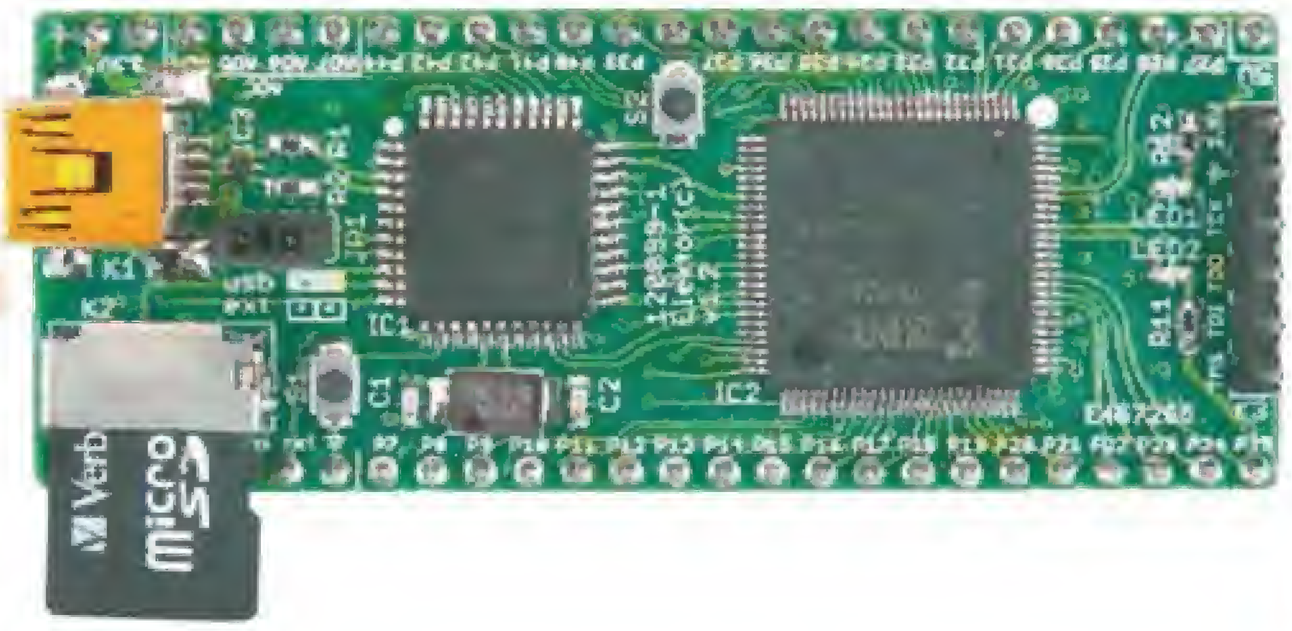
140 Minutes video presentation and more

5 DVD Feedback in Audio Amplifiers

In this Masterclass we address several aspects of feedback in audio amplifiers. The focus of this Masterclass, although not entirely math-free, is on providing insight and understanding of the issues involved.

Presenter Jan Didden provides a clear overview of the benefits that can be obtained by feedback and its sibling, error correction; but also of its limitations and

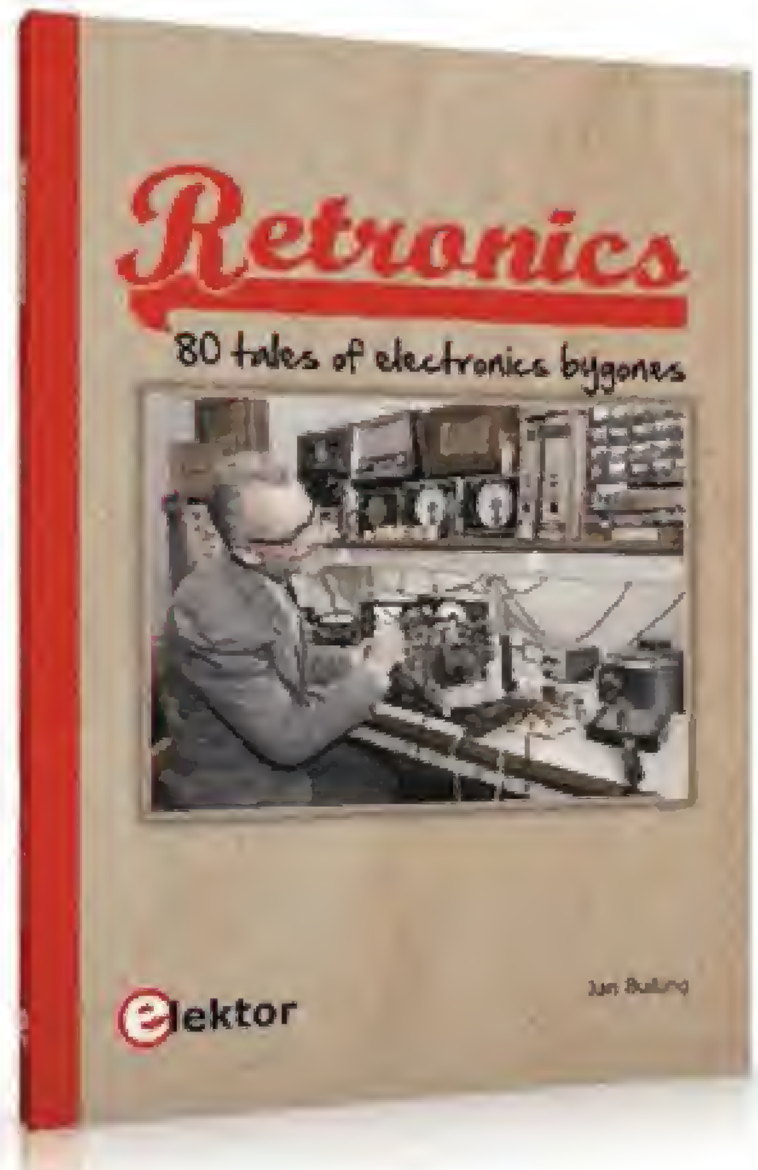
6



7

8

9



disadvantages. Recommended to audio designers and serious audio hobbyists!

ISBN 978-907920-16-5

£24.90 • € 29.95 • US $40.20

Taming the Beast

6 FPGA Development Board

FPGAs are unquestionably among the most versatile but complex components in modern-day electronics. An FPGA contains a maze of gates and other circuit elements that can be used to put together your own digital circuit on a chip. This FPGA development board (designed in the Elektor Labs) shows how easy it is for any electronics enthusiast, whether professional or amateur, to work with these programmable logic devices.

Module, ready build and tested Art.# 120099-91

See www.elektor.com/fpgaboard

Programming step-by-step

7 Android Apps

This book is an introduction to programming apps for Android devices. The operation of the Android system is explained in a step by step way, aiming to show how personal applications can be programmed. A wide variety of applications is presented based on a solid number of hands-on examples, covering anything from simple math programs, reading sensors and GPS data, right up to programming for advanced Internet applications. Besides writing applications in the Java programming language, this book also explains how apps can be programmed using Javascript or PHP scripts. When it comes to personalizing your smartphone you should not feel limited to off the shelf applications because creating your own apps and programming Android devices is easier than you think!

244 pages • ISBN 978-1-907920-15-8

£34.95 • € 39.95 • US $56.40

More than 75,000 components

8 CD Elektor's Components Database 7

This CD-ROM gives you easy access to design data for over 11,100 ICs, 37,000 transistors, FETs, thyristors and triacs, 25,100 diodes and 2,000 optocouplers. The program package consists of eight databanks covering ICs, transistors, diodes and optocouplers. A further eleven applications cover the calculation of, for example, zener diode series resistors, voltage regulators, voltage dividers and AMV's. A colour band decoder is included for determining resistor and inductor values. All databank applications are fully interactive, allowing the user to add, edit and complete component data.

ISBN 978-90-5381-298-3

£24.90 • € 29.50 • US $40.20

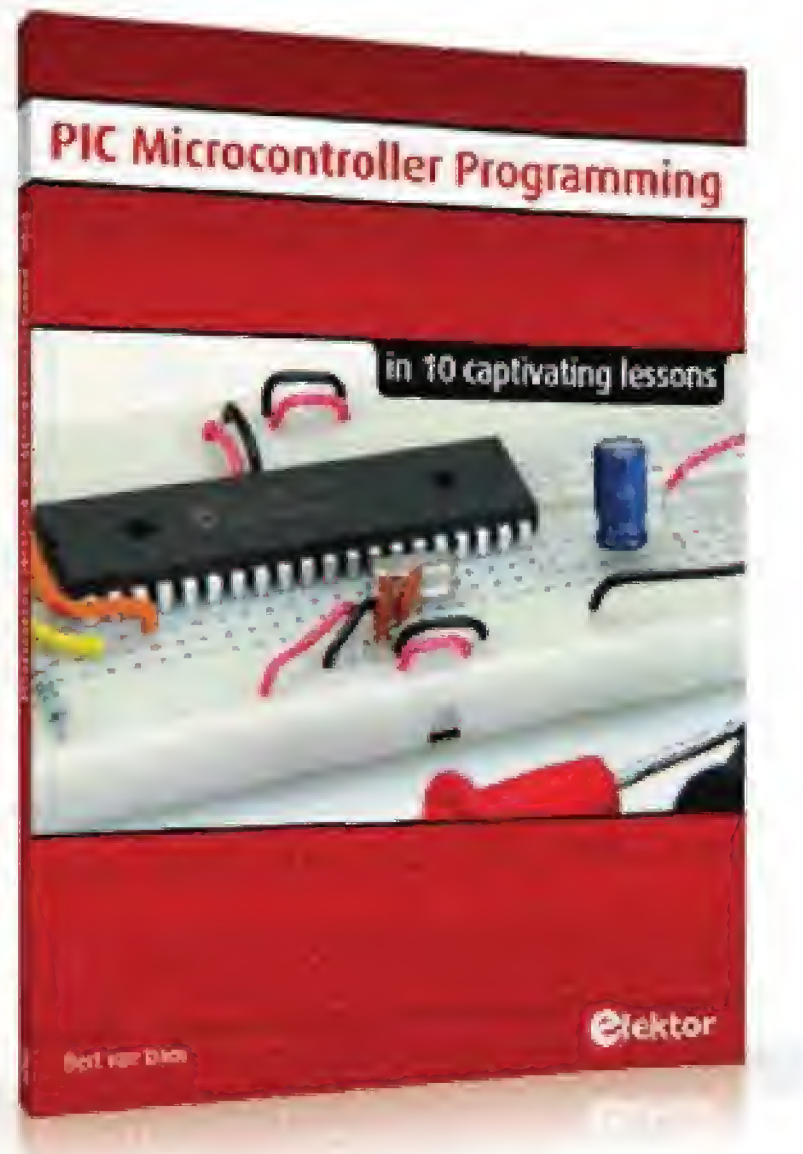
80 tales of electronics bygones

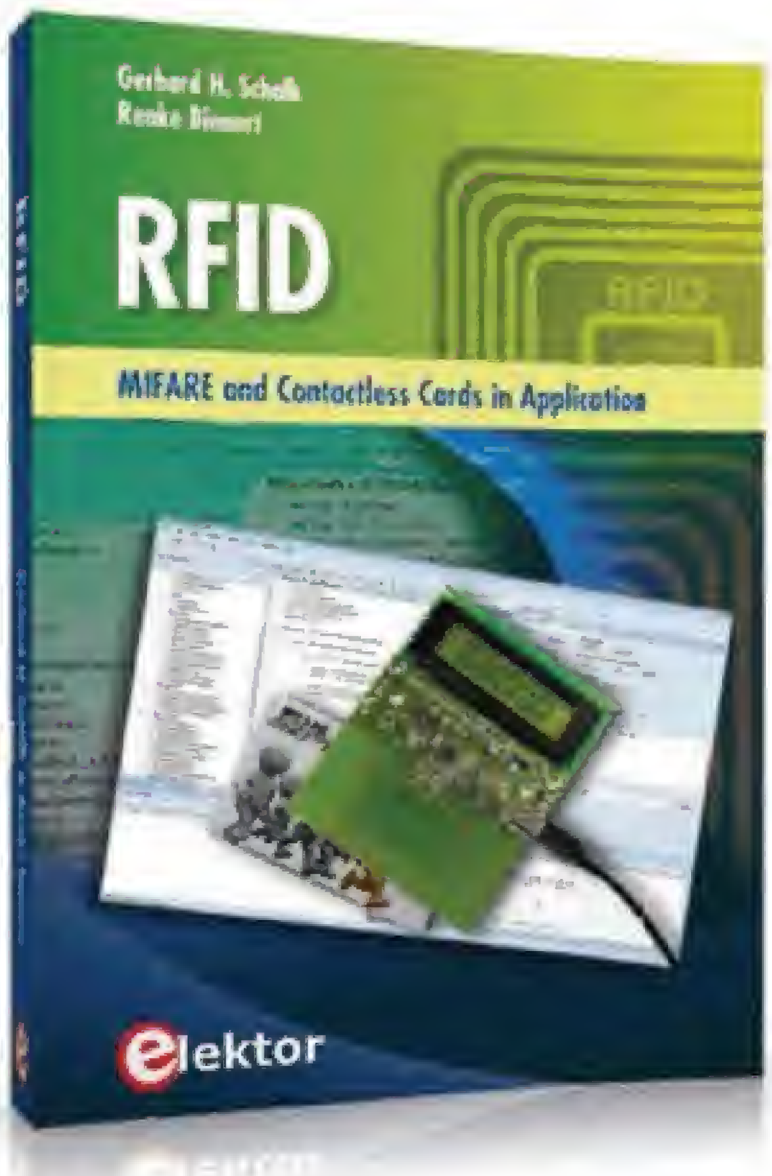
9 Retronics

Quite unintentionally a one-page story on an old Heathkit tube tester in the December 2004 edition of Elektor magazine spawned dozens of 'Retronics' tales appearing with a monthly cadence, and attracting a steady flow of reader feedback and contributions to the series. This book is a compilation of about 80 Retronics installments published between 2004 and 2012. The stories cover vintage test equipment, prehistoric computers, long forgotten components, and Elektor blockbuster projects, all aiming to make engineers smile, sit up, object, drool, or experience a whiff of nostalgia. Owners of this book are advised to not exceed one Retronics tale per working day, preferably consumed in the evening hours under lamp light, in a comfortable chair, with a piece of vintage electronic equipment close and powered up.

193 pages • ISBN 978-1-907920-18-9

£24.80 • € 29.95 • US $40.00





Wireless and button-free

Android Elektorcardioscope

Instructive, fascinating, and potentially useful to everyone: perform your own electrocardiograms on your Android smartphone or tablet! The project involves skillfully combining a small PIC interface to control an analog input stage with a great deal of software. Our ECG interface is available in the form of a ready-to-use module to which you just have to add four electrodes and an Android application for smartphone or tablet; there's no physical connection between this terminal and the interface, as it uses Bluetooth communication!

Ready assembled board • Art.# 120107-91
See www.elektor.com/elektorcardioscope

10 captivating lessons

PIC Microcontroller Programming

Using the lessons in this book you learn how to program a microcontroller. You'll be using JAL, a free but extremely powerful programming language for PIC microcontrollers. Assuming you have absorbed all lessons you should be confident to write PIC microcontroller programs, as well as read and understand programs written by other people. You learn the function of JAL commands such as include, pin, delay, forever loop, while loop, case, exit loop, repeat until, if then, as well as the use of functions, procedures and timer- and port interrupts. You make an LED blink, build a time switch, measure a potentiometer's wiper position, produce sounds, suppress contact bounce, and control the brightness of an LED. And of course you learn to debug, meaning: how to spot and fix errors in your programs.

284 pages • ISBN 978-1-907920-17-2
£29.50 • € 34.50 • US $47.60

MIFARE and Contactless Cards in Application

RFID

MIFARE is the most widely used RFID technology, and this book provides a practical and comprehensive introduction to it. Among other things, the initial chapters cover physical fundamentals, relevant standards, RFID antenna design, security considerations and cryptography. The complete design of a reader's hardware and software is described in detail. The reader's firmware and the associated PC software support programming using any .NET language. The specially developed PC program, "Smart Card Magic.NET", is a simple development environment that supports sending commands to a card at the click of a mouse, as well as the ability to create C# scripts. Alternatively, one may follow all of the examples using Visual Studio 2010 Express Edition. Finally, the major smart card reader API standards are introduced. The focus is on programming contactless smartcards using standard PC/SC readers using C/C++, Java and C#.

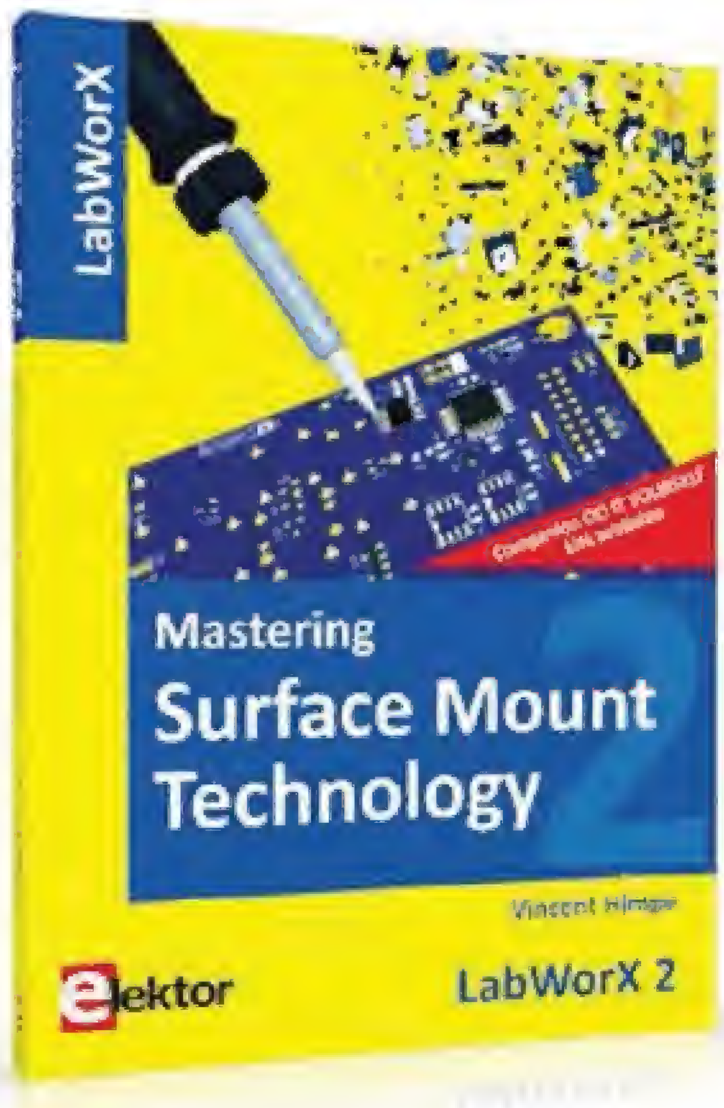
484 pages • ISBN 978-1-907920-14-1
£44.90 • €49.90 • US $72.50

110 issues, more than 2,100 articles

DVD Elektor 1990 through 1999

This DVD-ROM contains the full range of 1990-1999 volumes (all 110 issues) of Elektor Electronics magazine (PDF). The more than 2,100 separate articles have been classified chronologically by their dates of publication (month/year), but are also listed alphabetically by topic. A comprehensive index enables you to search the entire DVD. What's more, this DVD also contains the entire 'The Elektor Datasheet Collection 1...5' CD-ROM series, with the original full datasheets of semiconductors, memory ICs, microcontrollers, and much more.

ISBN 978-0-905705-76-7
£69.00 • € 89.00 • US $111.30



Ultrasensitive wideband E-smog detector

TAPIR Sniffs it Out!

Attention boy scouts, professionals and grandfathers! This ultrasensitive wideband E-smog detector offers you two extra senses to track down noise that's normally inaudible. TAPIR — short for Totally Archaic but Practical Interceptor of Radiation — also makes a nice project to build: the kit comprises everything you need. Even the enclosure, ingeniously consisting of the PCB proper! Using the TAPIR is dead easy. Connect the headphones and an antenna and switch it on. Move it around any electrical device and you'll hear different noises with each device, depending on the type and frequency of the emitted field.

Kit of parts, incl. PCB
Art.# 120354-71
£13.30 • € 14.95 • US $21.50

LabWorX 2

Mastering Surface Mount Technology

This book takes you on a crash course in techniques, tips and know-how to successfully introduce surface mount technology in your workflow. Even if you are on a budget you too can jumpstart your designs with advanced fine pitch parts. Besides explaining methodology and equipment, attention is given to SMT parts technologies and soldering methods. Many practical tips and tricks are disclosed that bring surface mount technology into everyone's reach without breaking the bank. A comprehensive kit of parts comprising all SMT components, circuit boards and solder stencils is available for readers wishing to replicate three projects described in this book.

282 pages • ISBN 978-1-907920-12-7
£29.50 • € 34.50 • US $47.60

Ideal reading for students and engineers

Practical Digital Signal Processing using Microcontrollers

This book on Digital Signal Processing (DSP) reflects the growing importance of discrete time signals and their use in everyday microcontroller based systems. The author presents the basic theory of DSP with minimum mathematical treatment and teaches the reader how to design and implement DSP algorithms using popular PIC microcontrollers. The author's approach is practical and the book is backed with many worked examples and tested and working microcontroller programs. The book should be ideal reading for students at all levels and for the practicing engineers who may want to design and develop intelligent DSP based systems. Undergraduate students should find the theory and the practical projects invaluable during their final year projects. Similarly, postgraduate students should be able to develop advanced DSP based projects with the aid of the book.

428 pages • ISBN 978-1-907920-21-9
£44.90 • € 49.90 • US $72.50

Further Information and Ordering: www.elektor.com/store
or contact customer service for your region

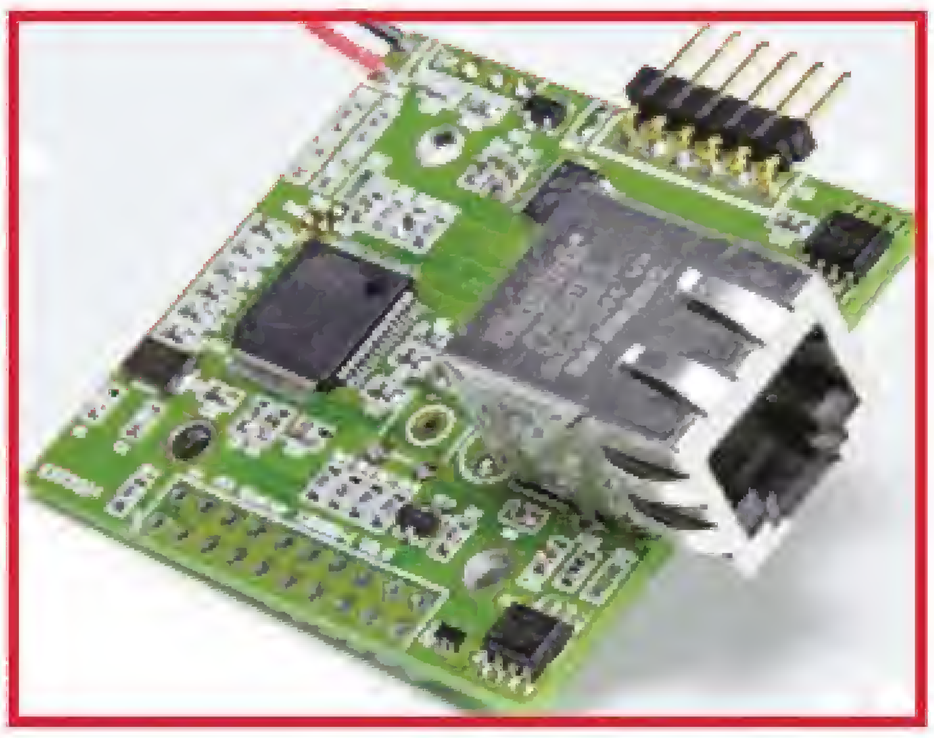
UK / ROW
Elektor International Media
78 York Street
London - W1H 1DP United Kingdom
Phone: +44 20 7692 8344
E-mail: service@elektor.com

USA / CANADA
Elektor US
111 Founders Plaza, Suite 300
East Hartford, CT 06108 USA
Phone: 860.289.0800
E-mail: service@elektor.com

NEXT MONTH IN ELEKTOR MAGAZINE

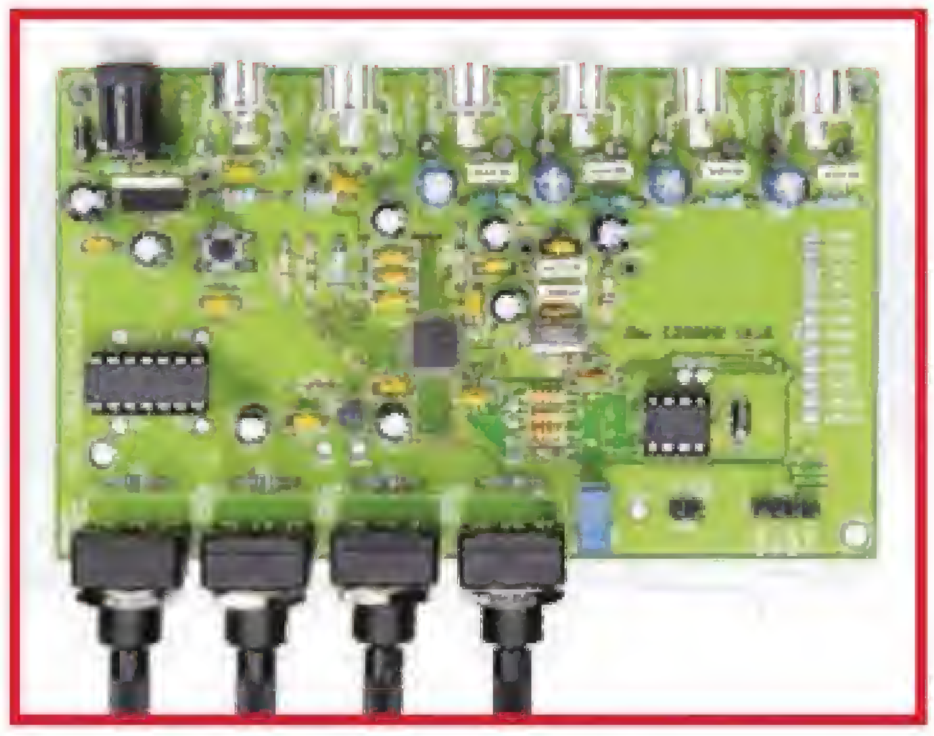
FPGA Extension Board

Since its publication back in December 2012 thousands of readers have used the Elektor FPGA board to delve into the possibilities of programmable logic. This tiny board does not have any peripheral circuitry, but that's redressed with the presentation of an expansion board that contains a comprehensive set of peripheral components such as an LCD, various buttons and LEDs, an A/D converter, an RGB color sensor, a pressure sensor, a temperature sensor and a GPS module.



Intelligent Network Tester

Faultfinding in an Ethernet network installed at home or in the office is not always easy. In such cases our Network Tester comes in handy. The design is compact and consists mainly of a single microcontroller and an Ethernet connector. Three LEDs on the board show whether a correct cable connection is available. The same for the presence of an Internet gateway, and if you are actually connected to the Internet. A good alternative to expensive commercial network analyzers!



DSP Entry-Level Board

If you want to start out with DSP you may be at a loss where to start and what to choose in terms of processor and software. This PCB offers a simple and affordable entry level opportunity into the DSP world. The board is set up around the type ADAU1701 from Analog Devices. Besides the DSP it contains just a few components including six RCA connectors for two audio inputs and four audio outputs, and four potentiometers for controlling the output signal levels.

Article titles and magazine contents subject to change, please check the Magazine tab at www.elektor.com

Elektor December 2013 is processed for mailing to US, UK and ROW Members starting November 12, 2013.

ORDERING INFORMATION

To order, contact customer service for your region:

USA / CANADA
Elektor US
111 Founders Plaza, Suite 300
East Hartford, CT 06108
USA
Phone: 860.289.0800
E-mail: service@elektor.com

Customer service hours:
Monday-Friday 8:30 AM-4:30 PM EST.

UK / ROW
Elektor International Media
78 York Street
London W1H 1DP
United Kingdom
Phone: (+44) (0)20 7692 8344
E-mail: service@elektor.com

Customer service hours:
Monday-Thursday 9:00 AM-5:00 PM CET.

PLEASE NOTE: *While we strive to provide the best possible information in this issue, pricing and availability are subject to change without notice. To find out about current pricing and stock, please call or email customer service for your region.*

COMPONENTS

Components for projects appearing in Elektor are usually available from certain advertisers in the magazine. If difficulties in obtaining components are suspected, a source will
normally be identified in the article. Please note, however, that the source(s) given is (are) not exclusive.

TERMS OF BUSINESS

Shipping Note:
All orders will be shipped from Europe. Please allow 2–4 weeks for delivery.

Returns
Damaged or miss-shipped goods may be returned for replacement or refund. All returns must have an RA #. Call or email customer service to receive an RA# before returning the merchandise and be sure to put the RA# on the outside of the package. Please save shipping materials for possible carrier inspection. Requests for RA# must be received 30 days from invoice.

Patents
Patent protection may exist with respect to circuits, devices, components, and items described in our books, magazines, online publications and presentations. Elektor accepts no responsibility or liability for failing to identify such patent or other protection.

Copyright
All drawings, photographs, articles, printed circuit boards, programmed integrated circuits, discs, and software carriers published in our books and magazines (other than in third-party advertisements) are copyrighted and may not be reproduced (or stored in any sort of retrieval system) without written permission from Elektor. Notwithstanding, printed circuit boards may be produced for private and educational use without prior permission.

Limitation of liability
Elektor shall not be liable in contract, tort, or otherwise, for any loss or damage suffered by the purchaser whatsoever or howsoever arising out of, or in connection with, the supply of goods or services by Elektor other than to supply goods as described or, at the option of Elektor, to refund the purchaser any money paid with respect to the goods.

MEMBERSHIPS

Membership renewals and change of address should be sent to the Elektor Membership Department for your region:

USA / CANADA

Elektor USA
P.O. Box 462228
Escondido, CA 92046
Phone: 800-269-6301
E-mail: elektor@pcspublink.com

UK / ROW

Elektor International Media
78 York Street
London W1H 1DP
United Kingdom
Phone: (+44) (0)20 7692 8344
E-mail: service@elektor.com

Do you want to become an Elektor GREEN or GOLD Member or does your current Membership expire soon?

Go to www.elektor.com/member.